ASG-Manager Products™ Controller's Manual

Version 2.5.1

Publication Number: MPR2100-251-CON
Publication Date: November 2001

ASG Worldwide Headquarters Naples Florida USA | asg.com | info@asg.com
1333 Third Avenue South, Naples, Florida 34102 USA Tel: 239.435.2200 Fax: 239.263.3692 Toll Free: 800.932.5536 (USA only)

ASG Documentation/Product Enhancement Fax Form

Please FAX comments regarding ASG products and/or documentation to (239) 263-3692.

Company Name	Telephone Number	Site ID	Contact name

Product Name/Publication	Version #	Publication Date
Product:		
Publication:		
Tape VOLSER:		

Enhancement Request:

ASG Support Numbers

ASG provides support throughout the world to resolve questions or problems regarding installation, operation, or use of our products. We provide all levels of support during normal business hours and emergency support during non-business hours. To expedite response time, please follow these procedures.

Please have this information ready:

- Product name, version number, and release number
- List of any fixes currently applied
- Any alphanumeric error codes or messages written precisely or displayed
- A description of the specific steps that immediately preceded the problem
- The severity code (ASG Support uses an escalated severity system to prioritize service to our clients. The severity codes and their meanings are listed below.)
- Verify whether you received an ASG Service Pack for this product. It may include information to help you resolve questions regarding installation of this ASG product. The Service Pack instructions are in a text file on the distribution media included with the Service Pack.

If You Receive a Voice Mail Message:

1. Follow the instructions to report a production-down or critical problem.
2. Leave a detailed message including your name and phone number. A Support representative will be paged and will return your call as soon as possible.
3. Please have the information described above ready for when you are contacted by the Support representative.

Severity Codes and Expected Support Response Times

Severity	Meaning	Expected Support Response Time
1	Production down, critical situation	Within 30 minutes
2	Major component of product disabled	Within 2 hours
3	Problem with the product, but customer has work-around solution	Within 4 hours
4	"How-to" questions and enhancement requests	Within 4 hours

ASG provides software products that run in a number of third-party vendor environments. Support for all non-ASG products is the responsibility of the respective vendor. In the event a vendor discontinues support for a hardware and/or software product, ASG cannot be held responsible for problems arising from the use of that unsupported version.

Business Hours Support

Your Location	Phone	Fax	E-mail
United States and Canada	800.354.3578	239.263.2883	support@asg.com
Australia	61.2.9460.0411	61.2.9460.0280	support.au@asg.com
England	44.1727.736305	44.1727.812018	support.uk@asg.com
France	33.141.028590	33.141.028589	support.fr@asg.com
Germany	49.89.45716.222	49.89.45716.400	support.de@asg.com
Singapore	65.6332.2922	65.6337.7228	support.sg@asg.com
All other countries:	1.239.435.2200		support@asg.com

Non-Business Hours - Emergency Support

Your Location	Phone	Your Location	Phone
United States and Canada	800.354.3578		
Asia	65.6332.2922	**Japan/Telecom**	0041.800.9932.5536
Australia	0011.800.9932.5536	**Netherlands**	00.800.3354.3578
Denmark	00.800.9932.5536	**New Zealand**	00.800.9932.5536
France	00.800.3354.3578	**Singapore**	001.800.3354.3578
Germany	00.800.3354.3578	**South Korea**	001.800.9932.5536
Hong Kong	001.800.9932.5536	**Sweden/Telia**	009.800.9932.5536
Ireland	00.800.9932.5536	**Switzerland**	00.800.9932.5536
Israel/Bezeq	014.800.9932.5536	**Thailand**	001.800.9932.5536
Japan/IDC	0061.800.9932.5536	**United Kingdom**	00.800.9932.5536
		All other countries	1.239.435.2200

ASG Web Site

Visit http://www.asg.com, ASG's World Wide Web site.

Submit all product and documentation suggestions to ASG's product management team at http://www.asg.com/asp/emailproductsuggestions.asp.

If you do not have access to the web, FAX your suggestions to product management at (239) 263-3692. Please include your name, company, work phone, e-mail ID, and the name of the ASG product you are using. For documentation suggestions include the publication number located on the publication's front cover.

Contents

Preface

This *ASG-Manager Products Controller's Manual* describes the capabilities and commands provided for the Controller of an ASG-Manager Products (herein called Manager Products) dictionary/repository. Manager Products is an integrated set of dictionary/repository-driven products developed for use on IBM System/370, 30*xx* and 4300 series, and plug-compatible machines.

Allen Systems Group, Inc. (ASG) provides professional support to resolve any questions or concerns regarding the installation or use of any ASG product. Telephone technical support is available around the world, 24 hours a day, 7 days a week.

ASG welcomes your comments, as a preferred or prospective customer, on this publication or on any Manager Products product.

About this Publication

This publication consists of these chapters:

- Chapter 1, "Introduction," gives you an introductory overview of the Controller's role in a Manager Products dictionary/repository.
- Chapter 2, "What Do You Want to Do?," asks you what tasks you want to complete and directs you to the relevant chapters or sections in this publication.
- Chapter 3, "Creating a Dictionary," overviews the tasks involved in dictionary creation.
- Chapter 4, "Security Facilities," describes the various ways in which you can provide security for the information contained in your dictionary.
- Chapter 5, "Logging," describes how to set up and manage a log of the dictionary commands processed and the benefits of establishing such a log. The chapter also describes the log's role in dictionary recovery.
- Chapter 6, "Dictionary Reorganization, Copying, and Backup," discusses how to reorganize a dictionary, how to copy a dictionary, how to take backups, and how to set up copies of various ASG-supplied dictionaries.

- Chapter 7, "Control of Member Locking," provides further information on the impact of member locking on your role as Controller.
- Chapter 8, "Status Facilities for Controllers," tells you how to set up and manage statuses (logical partitions) in your dictionary.
- Chapter 9, "User Defined Syntax for Controllers," provides an overview of how to define your own dictionary member types and member type structures and relationships, capabilities provided by the User Defined Syntax facility (UDS). A list of UDS Tables supplied by ASG is given.
- Chapter 10, "Miscellaneous Controller's Commands," lists various commands available to you as Controller or that have Controller's variants.
- Chapter 11, "The Master Operator," discusses the role of the Master Operator in a Manager Products installation.
- Chapter 12, "Commands for Dictionary Controllers," provides the specifications of every Controller's command or command variant. This chapter is intended as a reference chapter when you are using these commands, after you have familiarized yourself with the preceding descriptive chapters of this publication.

Publication Conventions

The following conventions apply to syntax diagrams that appear in this publication.

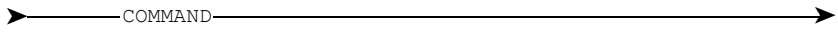
Diagrams are read from left to right along a continuous line (the "main path"). Keywords and variables appear on, above, or below the main path.

Convention	Represents
➤➤	At the beginning of a line indicates the start of a statement.
➤◄	At the end of a line indicates the end of a statement.
——➤	At the end of a line indicates that the statement continues on the line below.
➤——	At the beginning of a line indicates that the statement continues from the line above.

Keywords are in upper-case characters. Keywords and any required punctuation characters or symbols are highlighted. Permitted truncations are not indicated.

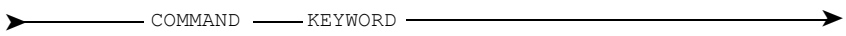
Variables are in lower-case characters.

Statement identifiers appear on the main path of the diagram:

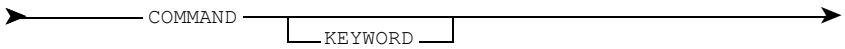


A required keyword appears on the main path:

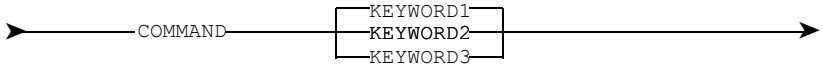
Convention Represents



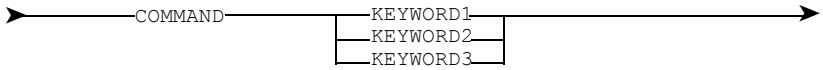
An optional keyword appears below the main path:



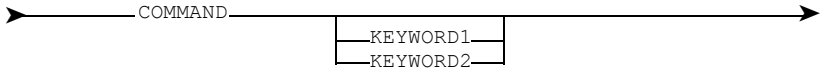
Where there is a choice of required keywords, the keywords appear in a vertical list; one of them is on the main path:



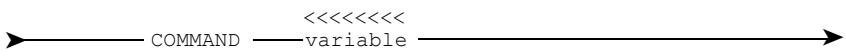
or



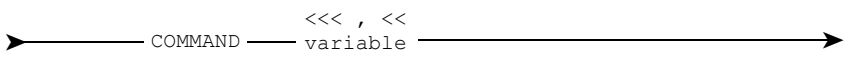
Where there is a choice of optional keywords, the keywords appear in a vertical list, below the main path:



The repeat symbol, <<<<<<, above a keyword or variable, or above a whole clause, indicates that the keyword, variable, or clause may be specified more than once:

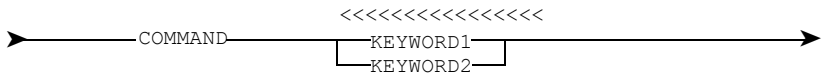


A repeat symbol broken by a comma indicates that if the keyword, variable, or clause is specified more than once, a comma must separate each instance of the keyword, variable, or clause:

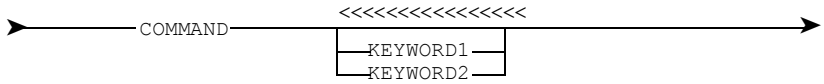


The repeat symbol above a list of keywords (one of which appears on the main path) indicates that any one or more of the keywords may be specified; at least one must be specified:

Convention	Represents



The repeat symbol above a list of key words (all of which are below the main path) indicates that any one or more of the keywords maybe specified, but they are all optional:



Allen Systems Group, Inc. uses these conventions in technical publications:

Convention	Represents
ALL CAPITALS	Directory, path, file, dataset, member, database, program, command, and parameter names.
Initial Capitals on Each Word	Window, field, field group, check box, button, panel (or screen), option names, and names of keys. A plus sign (+) is inserted for key combinations (e.g., Alt+Tab).
lowercase italic monospace	Information that you provide according to your particular situation. For example, you would replace *filename* with the actual name of the file.
Monospace	Characters you must type exactly as they are shown. Code, JCL, file listings, or command/statement syntax. Also used for denoting brief examples in a paragraph.
Vertical Separator Bar (|) with underline	Options available with the default value underlined (e.g., Y|N).

1 Introduction

This chapter details the Controller's duties and responsibilities.

Tasks Performed by the Controller

As dictionary Controller, you are responsible for creating dictionaries and controlling the security of all dictionary members.

Before you can use any Manager Products, you must enter them into the computer installation's software libraries. Your responsibility, as Controller, for instigating or maintaining standards can begin at, or before, the installation stage, when you must decide whether to tailor the Manager Products to conform to standards that have been devised.

Refer to the appropriate Manager Products installation publication for details of how to install Manager Products and for a summary of the installation macros which are used in the tailoring of Manager Products.

Note:

You can execute Manager Products in either a client/server (MPSF) mode or a non-client/server (standard) mode. This publication assumes the use of standard mode and only documents minor differences between the two modes. When executing under MPSF you can create repositories and MPAIDs using the Data-in-Virtual (DIV) access method. The DIV access method and significant MPSF installation and usage differences are documented in the *ASG-Manager Products Server Facility User's Guide*.

The dictionary must be created next with the CREATE/VCREATE command. Once a dictionary is created, it must be loaded with member definitions and related information. Individual definitions can be entered by any user by using the ADD command.

When working in a Manager Products dictionary, if the Automation of Set Up facilities for COBOL or PL/I (selectable units DMR-AS1 and DMR-AS2 respectively) are installed, large numbers of member definitions may be added to the dictionary from ANSI COBOL or PL/I source libraries in one operation. Where more than one dictionary is in use, you can SAVE dictionary information from an existing dictionary and RESTORE it into a new dictionary.

It is vital that you secure the information that has been loaded into the dictionary in two ways:

- Against unauthorized access
- Against corruption and loss

Prevention of access by unauthorized users is provided by means of user identities and passwords registered in the dictionary by a SECURITY command issued by the Controller. Only users who enter a registered password in an AUTHORITY command have dictionary access. You must register your own password (the master password) when you CREATE/VCREATE a dictionary.

You can also set up a Master Operator's password, which allows the user of this password to execute some of the Controller's private commands without having access to the dictionary's contents. You can thus delegate some of the dictionary maintenance tasks without having to disclose the master password. The Master Operator's password can be established or changed by using the Controller's AUTHORITY command. This command can also be used to change the master password.

The automatic recovery system provides security against corruption or loss of data. You can provide additional security by making regular backups using the UNLOAD command in conjunction with the logging facility or by using the backup facilities of the operating system.

It is your responsibility as Controller to establish stringent procedures for maintaining backup copies of the dictionary datasets and of any other important files. ASG recommends that the Controller make daily backups of all operational dictionaries, and retain, for example, one backup copy per week for a period of a month.

Note:

The Systems Administrator, acting as dictionary Controller, can also perform the dictionary Controller's functions described above, by using the Controller's commands.

For details of the job control requirements for Manager Products sessions that include any of the commands specified in this branch, please refer to the Manager Products installation publication appropriate to your environment.

Details of all Controller's commands mentioned in this chapter are given in Chapter 12, "Commands for Dictionary Controllers," on page 87.

2 What Do You Want to Do?

The purpose of this chapter is to direct Controllers who are unfamiliar with Manager Products to the chapters or sections relevant to the task they wish to perform. For each task the section and page reference in this publication is given.

Repository Controller's Tasks

Do you want to:

Establishing Security in a Repository

For an overview of the security system, refer to *ASG-Manager Products Dictionary/Repository User's Guide*.

Analyzing Repository Usage

Reorganizing/Copying a Repository

Reorganizing or copying a repository is a two-stage process: first you must copy what you want to save into an output dataset; then you must input the saved data/information into a new or an existing repository.

Backing Up/Recovering a Repository

Controlling Repository Usage/Processing

3 Creating a Dictionary

The creation of a dictionary involves these tasks:

- The allocation of disk space to contain the 5 datasets (files) that constitute the BDAM or VSAM-organized dictionary (or 4 datasets if logging is not to be applied to the dictionary).

  A DIV-organized dictionary always consists of a single physical dataset but is divided internally into the 4 or 5 components that comprise a BDAM or VSAM dictionary.

  Note:

  References to dataset in this publication should be interpreted to mean a physical dataset of a BDAM or VSAM-organized dictionary or a component of a DIV-organized dictionary.

- If either the Basic Status or Advanced Status facility is installed (selectable unit CMR-DD2 or CMR-AD21), the specification of the number of statuses required for the dictionary.
- If the Audit and Security facility (selectable unit CMR-DD3) is installed, the specification of the type of logging required.
- The pre-formatting of the 4 or 5 datasets (files) and their initialization with control information.
- The establishment of a master password to provide security in subsequent accesses to the dictionary.
- The loading of the datasets (files) with the dictionary information.

The last of these tasks can be performed by the Controller, acting upon information supplied by users and owners of the data; or it can be delegated to authorized users. The first 5 of the tasks are entirely in the province of the Controller.

The allocation of disk space to the dictionary and the naming of the 4 or 5 datasets it comprises is performed by the operating system. To achieve this, the Controller must write (or must authorize and provide the information for) job control statements in the job control language appropriate to the operating system for a first run of Manager Products. The job control requirements are discussed in the Manager Products installation publication appropriate to your environment.

The specification of the number of statuses and the type of logging required, the pre-formatting of the datasets that constitute the dictionary, their initialization with control information, and the establishment of a master password are achieved by means of the CREATE (BDAM or VSAM dictionary) or VCREATE (DIV dictionary) command. (The master password can be changed, if this later becomes necessary, by a special form of the AUTHORITY command).

The loading of the datasets with the dictionary information is done by means of UPDATE or ADD commands (or alternatively INSERT and ENCODE commands); and possibly by a SAVE of information from an existing dictionary to RESTORE into the new dictionary. Before any of these commands can be accepted by Manager Products, a DICTIONARY command followed by a valid AUTHORITY command must be input.

For details of the CREATE, SAVE, RESTORE, and AUTHORITY commands, refer to Chapter 12, "Commands for Dictionary Controllers," on page 87. For the ADD, UPDATE, INSERT, ENCODE, and DICTIONARY commands, please refer to *ASG-Manager Products Dictionary/Repository User's Guide*.

See *ASG-Manager Products Server Facility User's Guide* for details of the VCREATE command.

4 Security Facilities

This chapter includes these sections:

You are responsible for protecting the entire repository system against unauthorized access and for controlling access to restricted information held in the repository. You can do this by setting up these security measures:

- Security levels that apply to all members in the repository
- Passwords for users who are authorized to access the repository
- Security levels for specific users to restrict the access they have to members

Only you can access this security information.

Security for the Repository

If you have security functions you can use the SECURITY command to define, for all the members in the repository, these security levels:

- An *insertion* security level
- A *protection* security level
- A *lock-retention* period

You can protect the members in the repository by issuing an *insertion* security level for the repository. A user whose general security level is less than the insertion security level cannot insert or modify a member in the repository or remove a member from it.

You can control the members that a user is able to protect in the repository by issuing a *protection* security level for the repository. A user whose general security level is less than the protection security level cannot own any members.

If you have the workstation interface you can define how long members on the repository can remain locked by issuing a *lock-retention* period for the repository. The lock on any member expires at the end of this period.

Controlling User Access

Before any commands that process the repository are accepted, a user must be recognized by the password given in the AUTHORITY command. To enable a user to issue restricted commands, give them a Controller's or Master Operator's password using the AUTHORITY command.

Users with a Master Operator password can use unrestricted commands and some restricted commands, concerned with the physical backup and recovery of the repository and maintaining the repository's log. This enables you to delegate these routine tasks to other personnel if required.

Use the SECURITY command to define ordinary users in the repository and give them a user password so that they can issue unrestricted commands. When you define a user's password you can also define these security levels for users:

- A *general* security level
- A *specific* security level
- A user *lock-retention* period

If the *general* security level of a user is less than the security level of a member, the member is protected from that user. For example, if the security level of a user is 50 and the access security level of a member is 60, the user is unable to execute any command for that member.

To additionally define a *specific* security level, use the SECURITY command with the OWNER clause and define the user as an owner using the OWNER command. Typically an owner will be a department within the organization, but it might also be an individual systems analyst or a software development team.

Users become owners of members using the PROTECT command. Users can protect members from other users:

- By declaring themselves as an owner for a member
- By defining an access security level for a member
- By defining an alter security level for a member
- By defining a remove security level for a member

Any other user must have a general security level equal to or greater than the members. If you have the workstation interface facility, you can define how long locks applied by an individual user last before they expire.

Example of Controlling User Access

This example shows how access to members can be controlled using security levels.

User's Security Level

User	General Security Level PERSONNEL	Specific Security Level ACCOUNTS	Specific Security Level
PROGRAMMER	200	125	110
ANALYST	210	125	125
END USER1	100	—	90

The general security level of the users PROGRAMMER, ANALYST, and ENDUSER1 are compared with the specific security level of the owners PERSONNEL and ACCOUNTS when the users want to access owned members.

Members Security Levels

Member	Owned by	Access	Alter	Remove
FILE-EMP-MASTER	PERSONNEL	100	125	200
EMP-SAL	ACCOUNTS	100	125	200

The member EMP-SAL is owned by ACCOUNTS and the member FILE-EMP-MASTER is owned by PERSONNEL. To access, alter, or remove the members, a user's general security level must equal or exceed the security levels defined by the owner of the member.

Permitted Actions

User	EMP-SAL	FILE-EMP-MASTER
PROGRAMMER	Access	Access
	Alter	
ANALYST	Access	Access
	Alter	Alter
ENDUSER1	—	—

ENDUSER1 cannot access, alter, or remove EMP-SAL because ENDUSER1's specific security level in respect of the owner of EMP-SAL (ACCOUNTS) is less than the access, alter, and remove levels of EMP-SAL.

ENUDUSER1 cannot access, alter, or remove FILE-EMP-MASTER because ENDUSER1 has no specific security level in respect to PERSONNEL, the owner of FILE-EMP-MASTER.

5 Logging

This chapter includes these sections:

Introduction to Logging

The Manager Products logging facility provides a mechanism for recording commands issued in respect of a dictionary.

Logging can be applied to each Manager Products dictionary individually. ASG strongly recommends that logging be implemented on all dictionaries in order to obtain two major benefits:

- The ability to reconstruct the dictionary to its state at any desired point in time, by reapplying commands from the log to a dictionary reloaded from a backup copy
- The ability to analyze dictionary usage, to obtain information of value in the management of the dictionary and of the staff using it

The decision to apply logging to a dictionary is made by the Systems Administrator. Implementation would be a task for the dictionary Controller.

The log is established for a particular dictionary when the dictionary is created. If logging is to be applied to a dictionary that already exists without a log, the log can be established in two ways:

- By RELOADing a BDAM or VSAM-organized dictionary from a backup. For a DIV-organized dictionary you must use the second method given below.
- By RESTORing the dictionary into a dictionary CREATEd/VCREATEd with the AND LOG clause, if the dictionary needs to be rebuilt; that is, restructured or reorganized.

Once a log has been implemented for a particular dictionary, logging remains in force for the life of that dictionary. If logging is to be discontinued, the dictionary Controller should SAVE the dictionary and RESTORE it into a new dictionary CREATEd/VCREATEd without a log.

The log records information about transactions which update the dictionary. It is also possible to log all commands which are entered and to record messages.

An installation macro, DLOG is provided so that the logging facility can be tailored. The macro can be disregarded if no logging is to be applied.

These are the overheads involved in implementing a log for a dictionary:

- The disk space required for a log dataset
- A minimal increase in command execution time

The cost of these overheads is generally greatly outweighed by the greater physical security of the dictionary and the benefits that can be obtained from interpretation of the reports available. The variety of reports can provide all the information necessary for monitoring and controlling Manager Products use to any level of detail required.

The considerable benefits to be obtained from logging can only be fully realized if the log is effectively managed. This can be achieved by carefully planning and rigorously implementing the logging and backup procedures to be used at an installation. The procedures that are described for recovering a dictionary provide the dictionary Controller with the means to restore a dictionary provided that proper use is made of the facilities offered by logging.

Before establishing logging, ASG recommends that you familiarize yourself with Chapter 3, "Creating a Dictionary," on page 7 and Chapter 6, "Dictionary Reorganization, Copying, and Backup," on page 43, and with your Manager Products installation manual, which gives details of the installation macros and their keywords.

How the Log is Organized

The Organization of the Log

If logging is applied to a dictionary, the log is set up as a fifth dataset of that dictionary. For BDAM or VSAM-organized dictionaries, the name of the dataset is the dictionary name with the suffix J. The log dataset is preformatted, by the CREATE/VCREATE AND LOG or the RELOAD AND LOG command (for BDAM and VSAM only), in the same way as the other datasets of the dictionary.

The log dataset is logically subdivided into two equal areas:

- The primary area
- The secondary area

Logging takes place on a flip-flop basis between these areas. As one area is filled, logging switches automatically to the other. When an area becomes full, it is marked as AWAITING ARCHIVE. That is, a message is output to all users when they log on. The status of the log can be checked at any time by the dictionary Controller by issuing a LOG STATUS command which will output a report on the current status of the log. The area currently in use is marked as ACTIVE.

The filled area can be archived to magnetic tape by a LOG ARCHIVE command, after which it is marked as AVAILABLE, until the next switch takes place when it again becomes ACTIVE. An ACTIVE area cannot be archived. If both areas are allowed to become full, both are marked as AWAITING ARCHIVE, and further switching cannot take place until a LOG ARCHIVE command makes them AVAILABLE.

The rate at which the log dataset fills depends, among other things, on the size and number of transactions occurring.

If a LOG SWITCH command is used, it is possible to switch from an ACTIVE area to an AVAILABLE area.

See "Archiving the Log" on page 29 for details of the Log Status report.

See Chapter 12, "Commands for Dictionary Controllers," on page 87 for details of the LOG SWITCH command.

See "What is Logged" on page 17 for information on what is logged.

An Illustration of the Organization of the Log Dataset

When the log is created or after a LOG SWITCH UNCONDITIONAL command (followed by a LOG ARCHIVE command), the log is empty. One of the two areas is marked as ACTIVE. Transactions will be recorded on the ACTIVE area until this fills up, as illustrated in this table:

LOG Dataset

Primary Area	**Secondary Area**
Transactions Logged	Transactions Logged
0	0
ACTIVE	AVAILABLE

When it has become full, it then automatically becomes marked as AWAITING ARCHIVE. This is shown in the next table below. The secondary area then becomes ACTIVE. If both areas are allowed to become full then no further logging can take place until the log is archived.

No users will be able to issue any commands until the log containing transactions *n1* to *n14* has been archived.

LOG Dataset	
Primary Area	**Secondary Area**
Transactions logged	Transactions logged
n1, *n2*, *n3*, *n4*, *n5*, *n6*, *n7*, *n8*, *n9*, *n10*, *n11*, *n12*, *n13*, *n14*	0
AWAITING ARCHIVE	ACTIVE

After a LOG ARCHIVE command has been issued, the contents of the log dataset area, marked as AWAITING ARCHIVE, are written to the archive dataset. That area is then reset to ACTIVE. This table shows how a LOG ARCHIVE command has reset the primary area which was previously full:

LOG Dataset	
Primary Area	**Secondary Area**
Transactions logged	Transactions logged
0	*n15*, *n16*, *n17*, *n18*
AVAILABLE	ACTIVE

What is Logged

Logging Commands Which Update the Dictionary

If logging is implemented, all commands that update the dictionary are logged, irrespective of whether LOG UPDATE-COMMANDS or LOG ALL-COMMANDS has been specified.

For commands that have associated source input, such as ADD, MODIFY, REPLACE, INSERT, and ALTER, the associated source input (member definitions or amendment lines) is logged with the command.

For REMOVE *name-list* commands, and for ENCODE *name-list* and BULK ENCODE commands, each member processed is separately logged, as a separate transaction. Where a transaction results from the processing of one of the above commands, that fact is indicated in the log.

Commands that include an encoding phase, for example:

- ADD
- MODIFY
- REPLACE

generate two transactions in the log; the encoding is logged separately from the source record updating.

The logging takes place at the time the automatic error recovery system's update lock is set to off for each phase of the command processing. In the event of system malfunction any transaction that has taken place will have been written to the log and cleared from the error recovery dataset records.

For commands that update more than one member the error recovery system operates as though a separate command were issued for each member processed.

ASG-ControlManager (herein called ControlManager) commands are not logged. For example, the UPDATE command, provided by the Extended Interactive Facility (selectable unit CMR-FE01), is not logged. However, where such commands generate other commands that update the dictionary, those generated commands are logged. For example, ADD or MODIFY commands may be generated as a result of using the ControlManager Full Screen Editor in Update Mode.

PERFORM commands are not logged; however, commands generated by processing a PERFORM command are logged with the annotation VIA PERFORM.

As a general rule, commands that do not update the dictionary are not logged. An exception to this rule can occur with the CONSTRUCT UDS-TABLE commands. These are available with ControlManager User Defined Syntax facility (selectable unit code CMR-UD1) and are logged as update commands.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of commands that update the dictionary.

See *ASG-ControlManager User's Guide* for a description of how ControlManager Extended Interactive Facility commands are recorded.

See "The Log Analysis Report" on page 28 for details of what information is recorded about transactions in the reports available with logging.

Logging Commands Which Do Not Update the Dictionary

Commands which do not update the dictionary are not recorded on the log, unless LOG ALL-COMMANDS is specified. If it is specified, non-updating commands are recorded as enquiries. Commands that are recorded as enquiries include LIST, REPORT, WHAT, and others.

These commands are not logged, whether or not ALL-COMMANDS is stated or defaulted:

- AUDIT
- LOG
- CREATE
- VCREATE
- RELOAD
- DICTIONARY
- ROLL-FORWARD

Commands which update the MP-AID, such as EDIT, MP-AID DELETE, and others, are not logged.

Commands provided by the Extended Interactive Facility are not logged, but where such commands generate other commands that update the dictionary, those generated commands are logged.

Most CONSTRUCT commands are not logged.

An ADD, INSERT, or REPLACE command that fails because of a syntax error in the command itself or, in the case of ADD or INSERT, because the member named in the command already exists, is not logged as an update. This is because the command is rejected before an attempt is made to update the dictionary. If the log specification is ALL-COMMANDS, the failed command is logged as having failed. If one of these commands fails because of errors in the associated member definition, it is logged as an update, with result code 4.

Result codes are given in the Audit report for each transaction logged.

If LOG ALL-COMMANDS has been specified, then all commands issued are logged, apart from those mentioned above. See Chapter 12, "Commands for Dictionary Controllers," on page 87 for details of the LOG ALL-COMMANDS/UPDATE-COMMANDS command.

Message Logging

Message logging is available with the Audit and Security facility. It can be implemented across all dictionaries which have log datasets, by tailoring the installation macro DLOG. When this is done, the Systems Administrator can then specify that messages of particular severity levels (I, W, E, S, and/or C) and/or messages with particular identifying numbers are to be logged. Refer to your Manager Products installation manual for details.

If any messages that are specified to be logged (either by severity level or by number) are suppressed by a SWITCH OFF MESSAGES command, they continue to be logged if they occur.

If a message which is specified to be logged is generated by a non-updating command, when logging messages and updating commands only, then that non-updating command is also logged.

Messages which are output as a result of commands which cannot be logged, such as RELOAD, LOG, and DICTIONARY, cannot be logged.

Messages are not logged as separate transactions, but are recorded as they arise during the transaction logging the relevant command.

Refer to *ASG-Manager Products Messages Guide* for further information on messages.

Logging When Particular Selectable Units Are Installed

Auditing the Repository

You can produce audit reports to provide specific, detailed information, including:

- Who has been using the repository and what they have done
- What use was made of the repository at a particular time, if, for example, a security breach is suspected
- Which updating commands issued during a given period were unsuccessful (If a batch job was run overnight, this information could be used to avoid manual checking of the batch output.)
- What updates have been made to the repository and by whom
- Whether a particular user is using the repository efficiently, for example to monitor the progress of staff under training

You can use audit reports to recover selected parts of a log. For example, if a member is accidentally deleted, you can reapply the transaction that inserted the member.

If the roll-forward capability of the log has been inhibited, you can selectively reapply transactions to avoid the command that caused the problem.

You can use audit reports to help you make best use of available resources by picking out commands that take a long time to run.

The AUDIT command produces audit reports from information held on the repository's log or log archive dataset. If logging has not been implemented, an AUDIT command will fail. You can limit the report by selecting:

- Transaction numbers
- Commands
- Statuses
- The date and time

Audit reports include details of:

- Result codes
- EXCP counts (logical shared buffer accesses for a DIV dictionary)
- The environment and the CPU-ID

as well as the information provided by log status reports. See "The Log Status Report" on page 25 for details of log status reports.

If you have audit functions you can specify which messages and commands are to be logged for each repository.

See "Implementing Logging" on page 23 for details of implementing logging.

See *ASG-Manager Products Performance Tuning* for details of making the best use of your resources.

Logging with the User Interface Facility

For installations that include the User Interface facility (selectable unit CMR-UI1), commands are logged after they have been through any processing that may be performed in the input user exit.

Logging with the Status Facilities

If the Basic Status or Advanced Status facility (selectable unit CMR-DD2 and CMR-AD21, respectively) is installed, the name of the current status is included in the information recorded for each transaction in the log.

Logging with the User Defined Commands Facility

If the ControlManager User Defined Commands facility (selectable unit CMR-UD05) is installed and primary commands are renamed, then, subject to the UPDATE-COMMANDS or ALL-COMMANDS specification of the log, such commands are logged under their new names (that is, their new command identifiers). AUDIT string commands would select such commands under their new names, but the output would, subject to other selections in the commands, include any transactions that record the commands under their old names.

For example, if REMOVE has been renamed as DELETE, an audit of these commands would be initiated by:

```
AUDIT 'DELETE';
```

The output would include all transactions recording DELETE commands and also those recording earlier REMOVE commands.

Procedures for Implementing Logging

Decisions to Make Before Implementing Logging

The Systems Administrator is responsible for making decisions that affect all dictionaries at an installation, whereas the dictionary Controller is responsible for implementing the decisions applying to a specific dictionary.

When considering whether to implement logging, these decisions need to be made:

- To which dictionaries logging should be applied. ASG strongly recommends that logging should be applied to all active dictionaries, as it increases the physical security of the dictionaries and provides a valuable tracing mechanism in the event of human error or software malfunction.
- If the Audit and Security facility is installed:
  - — Individually for each dictionary to which logging is to be applied, whether to log all commands or only updating commands
  - — Generally for all dictionaries to which logging is to be applied, whether messages are to be logged; and if so, which messages or categories of messages
- In the light of the above decisions, whether to tailor the installation macro DLOG to log UPDATE-COMMANDS or ALL-COMMANDS. If neither of these keywords is stated, the value of the COMTYPE parameter of the DLOG installation macro (see your Manager Products installation manual) is taken as default.
- How to determine space allocations for the log datasets

Before establishing logging, ASG recommends that you familiarize yourself with Chapter 3, "Creating a Dictionary," on page 7 and Chapter 6, "Dictionary Reorganization, Copying, and Backup," on page 43, and with your Manager Products installation manual, which gives details of the installation macros.

Implementing Logging

When the decision as to whether to tailor the DLOG and DCUST installation macros have been made, the job control for Manager Products runs involving CREATE, VCREATE, UNLOAD, RELOAD, ROLL-FORWARD, and LOG commands can be set up or revised. Both these tasks are covered in the Manager Products installation manual appropriate to your environment.

If an existing BDAM or VSAM-organized dictionary for which logging has not been set up is to be moved over to logging, enter these commands:

```
UNLOAD ;
RELOAD dictionary name AND LOG selection ;
```

The RELOAD command can use only the output from an UNLOAD command. The AND LOG selection can be either ALL-COMMANDS or UPDATE-COMMANDS.

To establish a new dictionary with a log dataset, enter this command:

```
CREATE dictionary-name AND LOG selection ;
```

Alternatively, if implementing logging needs to be combined with dictionary reorganization or the dictionary access method is DIV, these commands are used:

```
SAVE ALL ;
CREATE dictionary name AND LOG selection ;
RESTORE ALL ;
```

The SAVE command backs up the dictionary, which can then be RESTOREd into a new, restructured dictionary with logging.

The AND LOG clause used with CREATE and RELOAD determines what will be recorded on the log. AND LOG UPDATES records only commands that update the dictionary. AND LOG ALL-COMMANDS logs all commands which are issued in respect of a dictionary and, optionally, messages generated as a result of these commands. The default value is AND LOG UPDATES. LOG ALL-COMMANDS is available only if the Audit and Security facility is installed (selectable unit CMR-DD3).

The initial decision can be changed at any time using the LOG ALL-COMMANDS. It cannot, however, be reversed. Once logging has been implemented, it cannot be discontinued. If logging is no longer required, the dictionary must be saved and then restored into a dictionary which is created without a log dataset.

Immediately after a CREATE, VCREATE, RELOAD, or RESTORE command (respectively, for the three cases mentioned above), an UNLOAD command should be issued to establish full roll-forward capability. This is achieved by an entry being made in the control record of the log dataset indicating that a backup has taken place.

Creating a New Log Dataset When the Log Dataset of a Dictionary is Unavailable

It may be necessary to open a dictionary in order to issue an UNLOAD command, when the log dataset of the dictionary is inaccessible, if, for instance, it has become corrupt. The LOG CREATE command accomplishes this by creating a new log dataset which is empty but with the correct last transaction number. Issuing this command allows the index, source, data entries, and error recovery datasets to be opened. When this command has been issued, the dictionary will close and a DICTIONARY command will have to be entered. The LOG CREATE command is not available for use with a DIV dictionary.

The UNLOAD command should then be issued to maintain full roll-forward capability.

Refer to Chapter 12, "Commands for Dictionary Controllers," on page 87 for details of the LOG CREATE command.

Physical Space for the Log Dataset

Once the decision to implement logging has been made, a log dataset must be established.

Unless overridden by a PHYSICAL-BLOCKSIZE or LOG-BLOCKSIZE clause in the CREATE command, the blocksize of the log dataset for a BDAM dictionary is set to 4096 bytes. For a VSAM dictionary, the blocksize of the log dataset is the CONTROLINTERVALSIZE minus 7.

ASG suggests using a blocksize of at least 4096 bytes for a log dataset. You should start with a reasonably high space allocation and adjust it later to fit your needs.

The log dataset is created when AND LOG is specified in the CREATE/VCREATE and/or RELOAD commands.

For a BDAM or VSAM-organized dictionary, you should allocate the log dataset on a different physical volume from the other four dictionary datasets. This ensures that full dictionary recovery is made or that full backup is reestablished in the event of failure of either physical volume.

Maintaining and Reporting the Log

Overview of Maintaining and Reporting the Log

The commands involved in log maintenance are LOG SWITCH and LOG ARCHIVE. LOG SWITCH is used to change manually the status of one of the areas of the log from AVAILABLE to ACTIVE. LOG ARCHIVE is used to archive to magnetic tape the contents of the primary area and/or the secondary area of the log dataset, depending on whether their status is AWAITING ARCHIVE.

The LOG STATUS command outputs a report on the status (ACTIVE, AVAILABLE, or AWAITING ARCHIVE) and utilization of the two areas of the log dataset, together with details of the last backup and roll-forward information.

Another command, LOG ANALYSIS, is available to output an analysis of transactions that have taken place and which have been written to the log. When this command is used, all or part of either the dictionary's log dataset or on the archived transactions is analyzed.

The LOG BACKUP-DETAILS and ROLL-FORWARD commands are provided to retain roll-forward capability if non-Manager Products backup software is used.

The ControlManager Audit and Security facility (selectable unit CMR-DD3) provides a further comprehensive selection of reports and analyses of the information contained in the log.

The Log Status Report

The Log Status Report— Log Status and Utilization Information

The tool provided by Manager Products for monitoring log usage in order to manage the log effectively is the Log Status report. This report is output by the LOG STATUS command.

The Log Status report includes this status and utilization information, for both the primary and the secondary areas of the log:

STATUS. There are two status types:

- AVAILABLE if an area is empty and is available for use when the active area is full (or when areas are switched by a LOG SWITCH command)
- AWAITING ARCHIVE if an area is full, or has been switched and can be made available only by writing its contents to the archive dataset (by a LOG ARCHIVE command).

FIRST TRANSACTION. This entry gives the transaction number, date, and time of the first transaction recorded in the area.

LAST TRANSACTION. This entry gives the transaction number, date, and time of the last transaction recorded in the area.

USAGE. This entry gives the percentage utilization of the area. If the area is AVAILABLE, usage is shown as 0 percent; if the area is AWAITING ARCHIVE, usage is shown as 100 percent.

FREE SPACE. This entry gives the space currently available in the area, in K bytes (K = 1024).

The dictionary Controller should regularly monitor the log and note the utilization of log space. Sufficient space should always be available to log further commands. If, by failing to archive in sufficient time, the dictionary Controller allows the log to become full, then all dictionary activity will cease until the log is archived. Regular inspection of the USAGE and FREE SPACE information output in the Log Status report for the two log areas will help the dictionary Controller avoid this.

The Log Status Report— Roll-Forward and Backup Information

The Log Status report also includes entries with roll-forward and backup information.

`LOGGING SELECTION`

A line is output in the Log Status report indicating whether LOG ALL-COMMANDS or LOG UPDATE-COMMANDS is implemented.

`LAST DICTIONARY BACKUP TAKEN`

The information on this line is generated automatically when an UNLOAD command is successfully actioned. If non-Manager Products software is used for a backup, this information is derived from the LOG BACKUP-DETAILS entry which should be input when such a backup is taken. Where no backup information is recorded, the word NONE is displayed. Dictionary roll-forward is not possible when no backup information is recorded.

`LOG CONTAINS RESULT CODE 8 TRANSACTION AT TRANSACTION` *`nnn`* `NO ROLL-FORWARD CAPABILITY BEYOND THIS RESULT CODE 8`

This line is included only if result code 8 transactions have been logged, with the resulting inhibition of roll-forward capability.

`ROLL FORWARD REQUIRES ARCHIVE DATASET`

This line, if present, indicates that a LOG ARCHIVE command has been successfully processed since the last recorded dictionary backup was taken; and that a roll-forward will require input from that command's output dataset. A subsequent LOG ARCHIVE command will also require that dataset as input.

`NO ROLL-FORWARD CAPABILITY`

If the dictionary's roll-forward capability has been inhibited, that is, if there are result code 8 transactions logged, or if neither an UNLOAD nor a LOG BACKUP-DETAILS command has been issued, the above message is output.

If result code 8 transactions are present in the log, the roll-forward capability is inhibited. A notification of their presence is, if appropriate, included in the Log Status report. If the roll-forward capability is inhibited, either because of the presence of result code 8 transactions in the log, or due to the absence of backup information in the log's control record (indicated by NONE in the LAST DICTIONARY BACKUP TAKEN line of the report), then this is also flagged in a line added at the end of the report. If the absence of roll-forward capability is flagged for either reason, a backup should be taken immediately.

Result codes are described in "Result Codes" on page 29.

Log Status Report—Information on Disk Accesses

The Log Status report also gives information on the number of disk input/output accesses (logical shared buffer accesses for a DIV dictionary) that would be required to roll-forward in the event of a system malfunction. This information is given in the ESTIMATED ROLL-FORWARD EXCPS entry of the report. It provides a basis for estimating the time required to perform a roll-forward.

If the EXCP count reaches or exceeds a number defined by the MAXEXCP parameter of the DLOG installation macro (see your Manager Products installation manual), a warning message is output each time the dictionary is opened.

The default value of MAXEXCP is 50000. When the defaulted or tailored value is approached or exceeded by the EXCP count, a dictionary backup should be taken. The EXCP count is reset to zero as a result of a successful UNLOAD command, or of a LOG BACKUP-DETAILS command following a non-Manager Products backup.

Note:

For a DIV repository, where no physical I/O is performed by MPSF clients, the default value or the value specified represents logical access to the shared buffer. You should specify a significantly larger value than you normally would specify for a BDAM or VSAM repository. For example, the default value of 50,000 should be increased to a value of no less than 2,000,000.

`ESTIMATED ROLL-FORWARD EXCPS`

This entry is omitted if no roll-forward capability exists. The line gives a count of the anticipated number of disk input/output accesses necessary to reconstruct the dictionary from its state at the time of the last backup to its current state.

The Log Analysis Report

The Log Analysis Report—Usage

The Log Analysis report (output by a LOG ANALYSIS command), is primarily used to determine the transaction to which a dictionary should be rolled forward, where the need to recover arose from a user error.

If loss of data resulted from a user error, the faulty transaction could be located by inspecting the Log Analysis report. A RELOAD AND ROLL-FORWARD TO TRANSACTION number command could then be issued. The transaction number specified in the RELOAD AND ROLL-FORWARD command would be the transaction number preceding the transaction number for the faulty transaction, as it appears on the Log Analysis report.

The Log Analysis report can also be interpreted to give some indications of whether non-optimized dictionary usage or a need for staff training exists. For example, sequences of repeated unsuccessful commands could indicate that there is a user who does not properly understand the syntax of the Manager Products language.

The Log Analyze Report—Information Recorded

The Log Analysis report contains detailed information of what is logged. For each transaction these entries are made:

TRANSACTION NUMBER. Each command logged (with its associated source input if applicable) is logged as a transaction with a sequential transaction number in the range 1 to 999999. After transaction number 999999 is logged, the numbering restarts at 1 for the next transaction.

START DATE. This gives the date and time a transaction was issued.

USER NAME. The identity of the user under whose authority the command was entered; that is, the dictionary user's name. If the password quoted is that of the Master Operator, the Log Analysis report output will not contain any details of transactions which involve allocation of passwords by the dictionary Controller. Thus, details of the AUTHORITY and SECURITY commands are suppressed. A message indicating that security details are not available to the Master Operator is output.

STATUS NAME. This contains details of the status name if the Basic or Advanced Status facility (selectable units CMR-DD2 or CMR-AD21, respectively) is installed, as well as these transaction details:

- The command input lines, with any associated source input (member definitions or amendment lines)
- The message number and any variable from any loggable message output by the command
- END DATE: The date and time the transaction ended

Result Codes

Result codes can be:

- 0 for a successful update
- 4 for an unsuccessful update
- 8 for a successful update that required input from an external file or the loading of an external table. These are the commands that, if successful, generate a result code 8:
  - RESTORE, CONVERT, and CONTROL UDS.

The presence of a result code 8 transaction in the log inhibits roll-forward capability for that transaction, and any subsequent transaction, until the next UNLOAD command. In installations in which all transactions are logged, result codes are logged only for updating commands.

For details of how to recover a dictionary if a result code 8 is present on the log, see "Recovering the Dictionary" on page 36.

Archiving the Log

Introduction to Log Archiving

The purpose of archiving is to make an area of the log dataset available for re-use, while preserving:

- Transactions that may need to be reapplied if a roll-forward becomes necessary
- A historical record of all dictionary activity from which dictionary Controllers with the Audit and Security facility installed can produce audit reports

A dictionary's log comprises its log dataset together with any archives taken from that dataset. The archive dataset itself has two functions:

- It is used as an input tape for merging with the contents of the log dataset
- The merged data forms an output dataset which can be used as input for a subsequent log archive

The name of the input archive dataset is the dictionary name with the suffix G. The input archive dataset is merged with the log dataset when a LOG ARCHIVE command is issued to produce a log output dataset.

The name of the output archive dataset is the dictionary name with the suffix H. This dataset is subsequently used as the input dataset for further LOG ARCHIVE commands. When a backup is carried out, a new archive dataset is created and the cycle starts again.

For maximum security, the archive dataset should be written to magnetic tape. Installations which have only one tape drive available should contact the ASG Service Desk for instructions on how to archive a log.

Note:

The input and output file definitions must always specify different datasets. Attempting to specify the same dataset will cause failure.

Log Switching When Archiving the Log

During archiving, the LOG SWITCH UNCONDITIONAL command can be used to ensure that both areas of the log are archived. If the command is issued, all the transactions on the log can then be archived in a single job-step, along the lines of the recommended procedures for archiving, regardless of the current status of the log. Both areas of the log dataset are marked as AWAITING ARCHIVE and can then be archived at the same time.

In order to ensure that both areas do not remain full, LOG SWITCH UNCONDITIONAL should only be used in a procedure that includes archiving. If both areas of the log are marked as AWAITING ARCHIVE, then no further logging is possible, and the dictionary becomes unavailable.

For details of what to do if a dictionary becomes unavailable because its log dataset is full, see "What to Do if a Dictionary is Unavailable and Its Log Dataset is Full" on page 41.

The Archive Cycle

The archive cycle refers to a routine procedure which has been set up for archiving the log dataset. If the archive cycle is firmly implemented, then the dictionary can be easily recovered, if necessary.

An archive cycle should commence immediately after each backup of the dictionary. The cycle is started automatically when an UNLOAD command is issued. If non-Manager Products software is used to take backups, the cycle should be started immediately after each backup by issuing a LOG BACKUP-DETAILS command.

When an UNLOAD command is processed, an entry is automatically made in the control record of the log dataset, recording when the command was issued. If a LOG BACKUP-DETAILS command is processed, an entry is similarly made in the control record of the log dataset, recording the information entered in the character string element of the command. Hence either of these commands can start an archive cycle, which is dependent on the information in the log dataset's control record. The control record includes information as to whether result code 8 transactions have been cleared, or the EXCP count has been reset to zero.

The frequency of archiving and the nature of the archive cycle followed depends on dictionary usage in a particular installation, and is a factor of:

- The log dataset size
- The number of transactions which take place over a certain period of time
- The presence of a result code 8 transaction

If the log dataset fills quickly, then regular archiving should take place to prevent the dictionary becoming unavailable.

If a high volume of transactions is regularly logged every day, then archiving should be set up as a dally routine, and carried out in the same job-step as backing up.

If result code 8 transactions are present, then a backup should be taken to allow full roll-forward capability.

The Archive Cycle Using Manager Products Backup Software

In this section it is assumed that UNLOAD commands are used to take backup copies of the dictionary. The SAVE command is used for reorganizing a dictionary, while UNLOAD is used for backing up a dictionary for these reasons:

- UNLOAD is quicker, because it makes a physical copy of a dictionary
- UNLOAD automatically resets the log, thereby starting an archive cycle
- UNLOAD is used in conjunction with RELOAD, which can be used with or without roll-forward

On acceptance of the first LOG ARCHIVE command following an UNLOAD command, a new output archive dataset is created. This archive dataset includes a control record containing information from the control record of the log dataset. All transactions from the AWAITING ARCHIVE areas of the log dataset are written to the archive dataset. The status of the archived area is reset to AVAILABLE.

Subsequent LOG ARCHIVE commands, up until the time the next UNLOAD is accepted, require the archive dataset as input. Transactions from the AWAITING ARCHIVE area of the log dataset are merged with the input archive dataset to produce an output archive dataset. The status of the archived area is reset to AVAILABLE.

With the next UNLOAD command, the cycle restarts. A new archive dataset is created by the next LOG ARCHIVE command (which does not require an input archive dataset).

ASG suggests that all transactions present on the log dataset should be archived immediately before a dictionary backup is taken. If this suggestion is followed, then all transactions logged between successive dictionary backups will be contained in a single archive dataset.

For additional security, the dictionary Controller should retain not only the latest UNLOAD dataset but at least one previous dataset.

In order to maintain roll-forward capability from a particular UNLOAD dataset, the dictionary Controller must keep a copy of the final output ARCHIVE dataset from every subsequent ARCHIVE cycle.

As an ACTIVE area cannot be archived, a LOG SWITCH UNCONDITIONAL command is needed preceding the LOG ARCHIVE command to ensure that the log is archived completely before the UNLOAD command is issued.

Recommended Procedure for Archiving Using Manager Products Software

In accordance with the recommended archive cycle, these steps would be taken:

- Opening the dictionary
- Disabling the dictionary to prevent access by other users
- Ensuring that all transactions will be archived by marking both areas of the log as AWAITING ARCHIVE
- Archiving all transactions on the log
- Taking the dictionary backup
- Enabling access by other users

To perform this operation, you would input this sequence:

```
DICTIONARY dictionary-name ;
AUTHORITY password ;
DISABLE ;
LOG SWITCH UNCONDITIONAL ;
LOG ARCHIVE ;
UNLOAD ;
ENABLE ;
```

The Archive Cycle Using Non-Manager Products Backup Software

It is necessary to record on the control record of the log dataset when the backup command was issued. Manager Products UNLOAD does this automatically, but non-Manager Products software does not, and a LOG BACKUP-DETAILS command must be issued when a backup has been completed.

You must use the Manager Products UNLOAD command to create a backup of a DIV dictionary created with a log. If you do not and it becomes necessary to rebuild the dictionary, then recovery of the dictionary from non-Manager Products backup will overwrite any transactions written to the log since the backup was taken and roll-forward of the dictionary to its latest updated state will not be possible.

On acceptance of the first LOG ARCHIVE command following a backup, a new output archive dataset is created. This archive dataset includes a control record containing information from the control record of the log dataset. All transactions from the AWAITING ARCHIVE areas of the log dataset are written to the archive dataset. The status of the area which has been archived is then reset to AVAILABLE.

Subsequent LOG ARCHIVE commands, up until the time the next backup is accepted, require the archive dataset as input. Transactions from the AWAITING ARCHIVE area of the log dataset are merged with the input archive dataset to produce an output archive dataset. The status of the archived area is reset to AVAILABLE.

With the next backup, the cycle restarts. A new archive dataset is created by the next following LOG ARCHIVE command (which does not require an input archive dataset).

ASG suggests that all transactions present on the log dataset should be archived immediately before a dictionary backup is taken. If this suggestion is followed, then all transactions logged between successive dictionary backups will be contained in a single archive dataset.

For additional security, the Controller should retain not only the latest backup but at least one previous one.

In order to maintain roll-forward capability from a backup, the dictionary Controller must keep a copy of the final output archive dataset from every subsequent archive cycle.

As an ACTIVE area cannot be archived, a LOG SWITCH command is needed preceding the LOG ARCHIVE command to ensure that the log is archived completely before the backup command is issued.

If a LOG SWITCH command is used, it is possible to switch from an ACTIVE area to an AVAILABLE area, provided that at least one transaction is recorded in the ACTIVE area. It would be used if an extensive update run required more space than is currently available on the log dataset.

Recommended Procedure for Archiving with Non-Manager Products Backup

Step 1 involves these tasks:

- Opening the dictionary
- Stopping access by other users
- Ensuring that all transactions will be archived, by marking both areas of the log as AWAITING ARCHIVE
- Archiving all transactions on the log
- Taking the dictionary backup

Step 2 involves these tasks:

- Opening the dictionary
- Recording details of the backup for subsequent display by the LOG STATUS command and activating the new archive cycle
- Recording details of the backup as a transaction on the log. The JOURNAL command used here effectively provides a permanent record of all non-Manager Products backups taken.

To perform this operation, you would input this command sequence for step 1:

```
DICTIONARY dictionary-name ;
AUTHORITY password ;
DISABLE ;
LOG SWITCH UNCONDITIONAL ;
LOG ARCHIVE ;
```

You would then carry out the non-Manager Products backup. On completion, you would then input this sequence for step 2:

```
DICTIONARY dictionary-name ;
AUTHORITY password ;
LOG BACKUP-DETAILS 'backup information' ;
JOURNAL 'backup information' ;
```

Maintaining Full Roll-forward Capability

Each transaction is assigned a number from 1 to 999999. Roll-forward refers to the application of a specific number or of all of these transactions from the log to the reloaded dictionary.

The roll-forward capability is the facility provided by logging to reapply transactions to a dictionary in sequence as they were originally issued so that the dictionary can be reconstructed if it becomes corrupted.

When full roll-forward capability does not exist, or is inhibited, a backup should be taken to restore it. Warning messages will be output at logon for all users, and an entry is also made in the Log Status report, which the dictionary Controller should regularly monitor.

Roll-forward is inhibited if:

- A result code 8 transaction occurs
- A specified or defaulted number of transactions (the MAXEXCP) has been exceeded, beyond which a roll-forward cannot take place
- If no backup details are recorded in the log

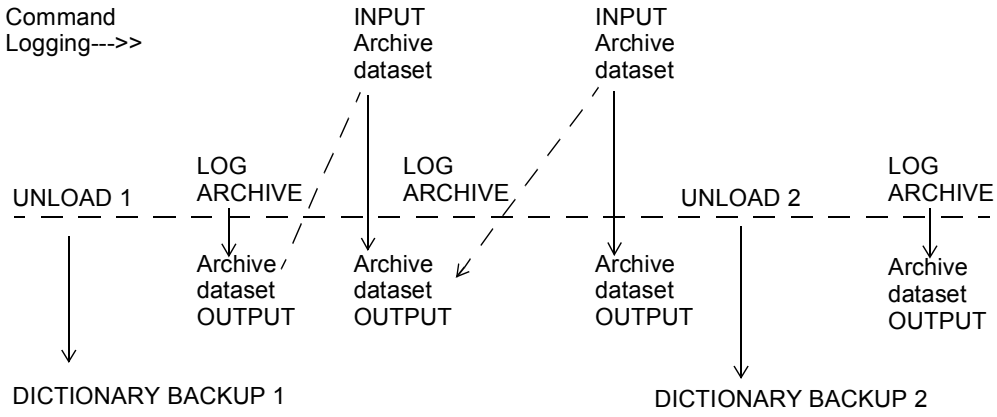
Whenever warning messages are output a backup should be taken.

The roll-forward process on recovering a dictionary normally operates from the last logged UNLOAD. However, a previous UNLOAD may be used if the subsequent transactions are still available on the log or archive datasets, or if the subsequent transactions do not need to be reapplied.

Illustrations of Possible Archive Cycles

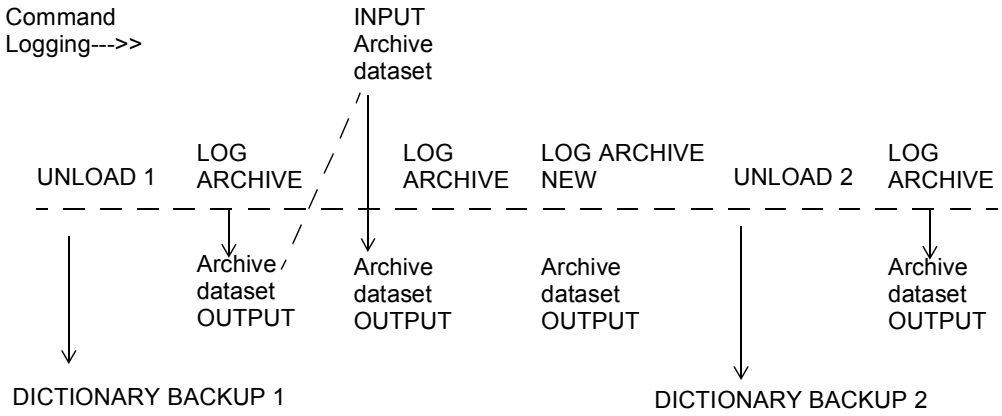
Scenario A:

Figure 1 • Scenario A—Dictionary backups are taken infrequently



The log dataset in Figure 1 is archived when a log area above is marked as AWAITING ARCHIVE. Dictionary backups are taken less often. Note that the OUTPUT archive dataset is used as the INPUT archive dataset for the subsequent LOG ARCHIVE command until a dictionary backup has taken place.

Figure 2 • Scenario B—A new archive dataset is created between backups



In Figure 2, dictionary backups are again taken infrequently. However, here a decision was made to use a new archive dataset tape. Note that a new output archive dataset is created.

Recovering the Dictionary

Introduction

Recovery refers to the process of restoring a dictionary and/or its data, if it becomes corrupted, or if data is lost through system malfunction or human error.

The dictionary Controller should plan in advance the procedure for routine recoveries and should keep careful notes while implementing a recovery, seeking advice from the ASG Service Desk should problems arise.

Recovery is discussed in this section in relation to a number of different circumstances.

Recovery with Result Code 8 Transactions

For each updating command logged, a result code is recorded. A result code 8 transaction is a successful update that could not be reapplied from the log alone, because it required input from an external dataset or the loading of an external table, and indicates that roll-forward will be inhibited.

The input associated with the logged command (unlike that associated with, for example, an ADD or a MODIFY command) is not copied to the log. These are the commands that, if successful, generate a result code 8:

- RESTORE
- CONVERT
- CONTROL UDS

These commands require data external to the dictionary which may no longer be available in the same form if it becomes necessary to reapply the command.

A backup should be taken immediately after any result code 8 transaction in order to reestablish a complete roll-forward capability.

If a transaction with a result code 8 is logged, an indicator is set in the log's control record. When this indicator is set, a roll-forward can proceed only as far as the transaction that generated the result code 8.

Some other procedure, such as manual recreation, could then be applied to recover the transactions (which must be purged before the dictionary can be opened). To recreate the dictionary manually, enter:

```
POST LOG ANALYSIS FROM transaction
DICTIONARY dictionary-name password ;
```

There are several possible areas where messages are output:

- The Log Status report indicates roll-forward inhibition
- The Log Analysis report indicates result codes for transactions which have been issued
- On issuing a DICTIONARY command, this message is output:

```
DN01043W ROLL-FORWARD CAPABILITY IS INHIBITED -
RESULT CODE B TRANSACTION SINCE LAST BACK-UP.
```

If the need to recover arises when roll-forward is inhibited by the presence of result code 8 transactions in the log, you should immediately contact the ASG Service Desk. Do not attempt to purge the log before contacting the ASG Service Desk. ASG will need to know the contents of POST report before being able to recommend manually recovering the dictionary.

If the dictionary Controller chooses to RELOAD and ROLL-FORWARD when the roll-forward is inhibited, then the LOG PURGE command must be applied after the partial roll-forward. The dictionary cannot be opened until the result code 8 and following transactions are purged.

Note:

The LOG PURGE command is not available for use with a DIV dictionary. Refer to the *ASG-Manager Products Server Facility User's Guide* for details of how to drop unwanted transactions from the log.

Recovery to a Particular Transaction

When the dictionary Controller wishes to back out a number of recent transactions from the dictionary; that is, he or she wants to restore the dictionary to its state before an undesired command was issued, one of two procedures is used.

Procedure 1. If there is an UNLOAD dataset that was created just before the transactions that are to be backed out, then this UNLOAD is identified. To rebuild the dictionary as of the time of the UNLOAD and to discard all subsequent transactions, enter:

```
RELOAD dictionary-name password NO-ROLL-FORWARD ;
LOG PURGE DICTIONARY dictionary-name password ;
```

Procedure 2. If the transactions to be backed out are after transaction *nnn*, then the last UNLOAD prior to transaction *nnn*, and any archives since that UNLOAD, are identified. Then to rebuild the dictionary as of the time of transaction *nnn*, and to discard all subsequent transactions, enter:

```
RELOAD dictionary-name password
AND ROLL-FORWARD TO TRANSACTION nnn ;
```

When the RELOAD processing is satisfactory (message DM12006I is output), enter:

```
LOG PURGE DICTIONARY dictionary-name password ;
```

The dictionary cannot be opened again until all transactions following a ROLL-FORWARD have been purged.

Note:

The LOG PURGE command is not available for use with a DIV dictionary. Refer to the *ASG-Manager Products Server Facility User's Guide* for details of how to drop unwanted transactions from the log.

Recovery When the Backup is Corrupted

This procedure can be used to recover the dictionary when the current backup tape or disk on which the most recent backup is held is damaged or unavailable:

1 Identify the previous available UNLOAD which can be used as the input to restore the dictionary and all archives since that UNLOAD.

2 Enter:

```
RELOAD dictionary-name password ;
```

3 The dictionary is first reloaded using the UNLOAD tape provided. The archive tapes are used in the provided sequence to roll-forward, applying all transactions in the order in which they were recorded in the current log dataset. The roll-forward is completed using the current log.

4 A sequence of messages such as these will show the status of the recovery:

```
DM122011 DICTIONARY HAS BEEN RELOADED

DM122501 INDEX LAST TCN nnnnn. AUTO ROLL-FORWARD INVOKED

DM122641 ROLL-FORWARD HAS OPENED ARCHIVE. START TCN IS nnnnn.
         END TCN IS nnnnn.

DM122541 ROLL-FORWARD IN PROGRESS

DM122561 TRANSACTION nnnnn HAS BEEN APPLIED

*

DM12256I TRANSACTION nnnnn HAS BEEN APPLIED

DM12265I ROLL-FORWARD HAS CLOSED ARCHIVE

*

DM12264I ROLL-FORWARD HAS OPENED ARCHIVE. START TCN IS nnnnn.

         END TCN IS nnnnn.

DM12254I ROLL-FORWARD IN PROGRESS

DM12256I TRANSACTION nnnnn HAS BEEN APPLIED

*

DM12256I TRANSACTION nnnnn HAS BEEN APPLIED

DM12265I ROLL-FORWARD HAS CLOSED ARCHIVE
```

5 These messages displays until all but the last archive tape has been rolled forward, then you will see messages such as these:

```
DM12264I ROLL-FORWARD HAS OPENED ARCHIVE. START TCN IS nnnnn.

         END TCN IS nnnnn.

DM12254I ROLL-FORWARD IN PROGRESS

DM12256I TRANSACTION nnnnn HAS BEEN APPLIED

*

DM12256I TRANSACTION nnnn HAS BEEN APPLIED

DM12255I ROLL-FORWARD HAS ENDED

DM12006I RELOAD PROCESSING SUCCESSFUL
```

If the roll-forward encounters an out-of-order transaction number (presumably indicating an archive tape/disk mounted out of sequence), then one of these messages is issued:

```
DM12259E TRANSACTION nnnnn NOT PRESENT ON dataset

M12503E  TRANSACTION CONTROL NUMBERS ARE NOT CONSECUTIVE
```

and the roll-forward is halted. Identify the correct archive and enter:

```
ROLL-FORWARD dictionary-name password ;
```

Recovery Using Non-Manager Products Backups

Although ASG does not recommend it, you can use non-Manager Products software to take backups. When taking a backup of a BDAM or VSAM-organized dictionary using non-Manager Products software, the Controller should ensure that the log dataset, if present, is not dumped as part of the backup process.

While the log dataset is an integral part of the dictionary, it represents a record of commands issued since a backup, which may be required as input to a roll-forward operation. By dumping the log dataset with the four other datasets there is always the danger that a restore operation will overwrite the log dataset and so render roll-forward impossible.

If the log dataset is allocated on a different physical volume from that of the other four datasets, then even if the log dataset is dumped as part of the backup process and a physical disk restore operation (that is a non-Manager Products restore operation) is employed, the log dataset will not be overwritten.

If non-Manager Products backups are employed, full recovery of a dictionary requires a ROLL-FORWARD command to be issued following the non-Manager Products restore operation. ASG suggests that the output be carefully examined as soon as the roll-forward is complete, to determine that the recovery is correct; and that a fresh backup be taken. If non-Manager Products software is again used for this backup, a LOG BACKUP-DETAILS command should be issued as soon as the backup is completed to record the backup in the control record of the log dataset.

You must use the Manager Products UNLOAD command to create a backup of a DIV dictionary created with a log. If you do not and it becomes necessary to rebuild the dictionary, then recovery of the dictionary from non-Manager Products backup will overwrite any transactions written to the log since the backup was taken and roll-forward of the dictionary to its latest updated state will not be possible.

What to Do if a Dictionary is Unavailable and Its Log Dataset is Full

If both areas are AWAITING ARCHIVE, the dictionary becomes unavailable and this message is output:

```
DM00066S DICTIONARY NOT AVAILABLE
```

Until the dictionary Controller archives the log, further logging is not possible and no commands from users other than the Controller are accepted. The Controller will not be able to issue updating commands, but can issue non-updating commands, even if logging has been set up to LOG ALL-COMMANDS. However, no commands can be logged while the log remains full.

If this situation occurs, the Controller should immediately issue these commands to archive and then unload the dictionary:

```
CLOSE ;
DICTIONARY dictionary-name  ;
AUTHORITY password;
LOG SWITCH UNCONDITIONAL ;
LOG ARCHIVE ;
UNLOAD ;
```

The dictionary can now be accessed with the usual DICTIONARY and AUTHORITY commands, and again becomes available to all users.

6 Dictionary Reorganization, Copying, and Backup

This chapter includes these sections:

Dictionary Reorganization and Copying

From time to time it may become necessary for you to reorganize a dictionary. It is sometimes necessary to do so when changing from one release of Manager Products to another; the Migration Notices issued with each release state whether or not reorganization is necessary. The need to reorganize may also arise if it is found that the blocksize previously selected for any dataset needs to be changed. The ANALYZE command is provided to assist you, the Controller, to determine whether this is the case.

The reorganization of a dictionary for either of the above-mentioned reasons involves your creating a physically new dictionary, then transferring of the contents of the old dictionary into the new.

Occasionally you may need to:

- Copy part of the contents of one dictionary into another (for example, where a master and separate project dictionaries are maintained)
- Copy a dictionary created under DOS for use under OS
- Copy a dictionary created under OS for use under DOS

You can use the Controller's private commands, SAVE and RESTORE, to achieve any of these.

The SAVE command achieves a partial reorganization of the saved data; that is, due to previous alterations, information relating to a member is fragmented (held in different parts of the dataset). Such information is brought together when written to the output dataset.

If a further DICTIONARY command reopens a dictionary during the same session, any SAVE command entered after the dictionary has been reopened will overwrite any output from a SAVE command written when the dictionary was opened earlier.

The output dataset from a SAVE command is intended only for subsequent use with a RESTORE command. Its format is such that it is of no use in any other context.

The RESTORE command completes the reorganization of saved data. The restored records are written in the block sizes specified when the receiving dictionary was CREATEd or VCREATEd in the case of a DIV dictionary.

Because restoring a BDAM or VSAM-organized dictionary can be a lengthy process (depending on the size of the dictionary) the RESTORE ALL command is restartable. That is, if, for any reason, the RESTORE command fails to complete the dictionary reorganization successfully, the command can be restarted from the point at which it failed. For a DIV dictionary, you cannot restart the RESTORE command.

Only one dictionary is open at a time; so if you need to copy material from one dictionary to another, a DICTIONARY command and an AUTHORITY command (quoting the Controller's password) must precede the SAVE command, and another DICTIONARY command and AUTHORITY command (quoting the Controller's password) must precede the RESTORE command.

If you are copying material to and from the same dictionary during the same Manager Products session, only one DICTIONARY command and one AUTHORITY command are needed.

Dictionary Backup

For taking copies of a dictionary for backup purposes, two further Manager Products commands are provided: the UNLOAD and RELOAD commands. (The SAVE and RESTORE commands together perform a reorganization of the dictionary. ASG therefore recommends that these commands be reserved for reorganizing and copying a dictionary.)

The UNLOAD and RELOAD commands can be used together with the logging facility when copying a dictionary to reconstitute the latest updated state of the dictionary.

The automatic recovery system provides a further level of safeguard against corruption or loss of data arising from interruption of a Manager Products session, without the need to reload and roll-forward from the last backup.

For a BDAM or VSAM-organized dictionary, the log dataset should be allocated on a different physical volume from the other 4 dictionary datasets. This will ensure that, if there is any failure of the disk volume containing the log dataset, a dictionary with full backup can be reestablished (if the need to do this arises, contact the ASG Service Desk for instructions); and that, if there is any failure of the disk volume containing the other 4 dictionary datasets, recovery can be made to any logged transaction (by a RELOAD command with roll-forward).

ASG recommends that you use UNLOAD in preference to other non-Manager Products dump utilities, because:

- Only Manager Products software can ascertain which of the blocks (established when you CREATEd the dictionary) have been utilized. Unused blocks are not dumped. The UNLOAD command is therefore quicker than other utilities when dumping a partially utilized dictionary.
- A RELOAD from an UNLOAD dataset can automatically invoke roll-forward to reconstitute the dictionary to any requested transaction present on the log.
- Manager Products software ensures that the dictionary cannot be updated while an UNLOAD command is being processed.

You must use the Manager Products UNLOAD command to create a backup of a DIV dictionary created with a log. If you do not and it becomes necessary to rebuild the dictionary, then recovery of the dictionary from non-Manager Products backup will overwrite any transactions written to the log since the backup was taken. Roll-forward of the dictionary to its latest updated state is not possible.

If, under CMS or OS, the operating system's dump utility is used to take backup copies, you should ensure that the dictionary cannot be updated during the copying process. You can do this by including DISP= OLD in the job control statements for the 4 or 5 dictionary datasets.

DOS users should not use their operating system's dump utility, as it is not suitable for taking dictionary backups. (DOS dictionary datasets have user labels.)

The UNLOAD command requires DICTIONARY and AUTHORITY commands to have been issued. For an UNLOAD command to be accepted, the password quoted in the AUTHORITY command can be either that of the Controller or that of the Master Operator. A RELOAD command does not require preceding DICTIONARY and AUTHORITY commands. (In this respect it is similar to CREATE/VCREATE.) The master password or Master Operator's password of the UNLOADed dictionary is stated instead in the RELOAD command.

Changing Dictionary Space Allocation

To determine the disk space utilization in a dictionary, use the QUERY DICTIONARY command.

You can increase or decrease the space allocated to a BDAM or VSAM dictionary's datasets, without having to reorganize the dictionary, by using UNLOAD followed by a RELOAD. You should specify the required space allocation in the JCL statements used by the RELOAD command and then RELOAD the dictionary into a newly allocated area of storage. Use the SAVE and RESTORE command for a DIV dictionary.

If you want to change the space allocation and reorganize the dictionary, use the SAVE and RESTORE commands instead.

(The processing of UNLOAD and RELOAD is appreciably faster than that of SAVE and RESTORE.)

To determine the disk space utilized by specific members, or types of member, in a dictionary, use the ANALYZE command.

If you want to change the blocksize of any of the datasets, you should use the CREATE/VCREATE command.

Restoring ASG-supplied Dictionaries

The Manager Products installation manual relevant to your environment gives details of how to set up your Manager Products Administration Dictionary, together with the job control statements needed to restore ASG-supplied dictionaries into that dictionary. The information given here is in amplification of that given in your installation manual.

The Manager Products Administration Dictionary should act as host to:

- The UDS Table Dictionary, where the User Defined Syntax facility (selectable unit CMR-UD1) is installed
- The InfoDictionary, where the User Defined Infosystem facility (selectable unit CMR-UD10) is installed
- The DictionaryManager Translation Rules Dictionary, where the Corporate Dictionary Definition Export for IDD facility (selectable unit DYR-TE08) is installed

To restore dictionaries from the installation tape into the Manager Products Administration Dictionary

1 Sign on to the Manager Products Administration Dictionary as that dictionary's Controller.

2 If you have the User Defined Syntax facility, you must restore the UDS Table Dictionary into the empty Manager Products Administration Dictionary. Use the command:

```
RESTORE ALL ;
```

As well as restoring the members from the UDS Table Dictionary, this leaves the UDS table provided by the Manager Products load module DU007 as the UDS table that is operative in the Manager Products Administration Dictionary.

3 Where the Basic Status or Advanced Status facility is installed (selectable unit CMR-DD2 or CMR-AD21), ASG recommends that separate statuses be maintained for each ASG-supplied dictionary within the Manager Products Administration Dictionary. A separate status should also be maintained for Profile members (LOGON-PROFILE and GLOBAL-PROFILE member types), for security reasons. At this stage, therefore, you should name the statuses within your dictionary using STATUS NAME commands. (See Chapter 12, "Commands for Dictionary Controllers," on page 87 for details of the STATUS NAME command.)

Note:

If you have restored the UDS Table Dictionary into the Manager Products Administration Dictionary, the first status will already have been named UDS. You can rename this status if you wish.

4 If the User Defined Infosystem facility is installed, issue the commands:

```
STATUS udi-status-name ;
RESTORE SOURCE ;
BULK ENCODE UNVERIFIED ;
```

where the first of these commands, quoting the name you have given to the InfoDictionary status, is issued only if the Status facility is installed. Omit all three commands if the User Defined Infosystem facility is not installed.

The restored InfoDictionary will not include the INFOBANK-PANEL members relating to Manager Products and selectable units that are not included in your Manager Products configuration. When your InfoDictionary is encoded, therefore, it will include a number of dummy members, resulting from references in the menu paths and other selection paths to the excluded members.

5 If the Corporate Dictionary/Repository Definition Export to IDD facility (selectable unit DYR-TE08) is installed, you must restore the ASG-DictionaryManager Translation Rules Dictionary into the empty Manager Products Administration Dictionary, by entering these commands:

```
STATUS te08-status-name ;
RESTORE SOURCE ;
BULK ENCODE UNVERIFIED ;
```

where the first of these commands (quoting the name you have given to the Translation Rules Status) is issued only if the Status facility is installed. Omit all three commands if the Corporate Dictionary/Repository Definition Export to IDD facility is not installed.

The job control for the operations described above is given in the Manager Products installation manual relevant to your environment.

Restoring the DEMO Dictionary

Manager Products release tapes include a dataset called MP.DEMO, which contains the dictionary (named DEMO) used for the Manager Products Demonstration and Training Guide. MP.DEMO is the output of a SAVE ALL command issued on the DEMO dictionary. The Controller can retrieve the DEMO Dictionary by issuing a RESTORE ALL command.

Details of the release tapes, and examples of job control for retrieving the DEMO dictionary, are given in the Manager Products installation manual appropriate to your environment. The release tape also includes outline job control and Manager Products commands for retrieving the DEMO dictionary. Details of the disk space to be allocated to the DEMO dictionary are given in the appropriate installation manual.

The dictionary into which the example dictionary is restored should have been created by the command:

```
CREATE dict MASTER 'password'
INDEX xxx SOURCE xxx DATA xxx WITH xxx STATUSES ;
```

The three statuses DMR, CONV, and TEST are supplied with encoded members. In other named statuses, members are supplied unencoded. Before one of these other statuses (except status UDS) is used, these commands should be issued:

```
DICTIONARY dict ;
AUTHORITY 'ANALYST' ;
STATUS status-name ;
BULK ENCODE ;
```

These commands can be issued by the Controller or by any user.

If the User Defined Syntax facility (selectable unit CMR-UD1) is installed, then before the DEMO dictionary is used, the Controller must issue these commands:

```
DICTIONARY dict ;
AUTHORITY 'password' ;
CONTROL UDS DUO12 ;
STATUS UDS ;
BULK ENCODE;
```

See "CONTROL UDS" on page 132 for details of the CONTROL UDS command.

For installations that do not include the Basic Status or Advanced Status facility (selectable unit CMR-DD2 or AD21), the following sequence of commands will make 1 status of the example dictionary available for use. Either of the statuses CONV or UDS can be utilized.

- Status CONV uses Manager Products' basic member type set.
- Status UDS uses the Extended Data Processing Structure (EDPS) member type set, one of the alternative member type sets supplied by ASG with the User Defined Syntax facility: it is therefore available only if that facility, selectable unit CMR-UD1, is included in the installation.

Whichever of these statuses is selected for use, status DMR must also be restored; because both status CONV and status UDS, in the multi-status DEMO dictionary, look back to the frozen status DMR for definitions at the GROUP and ITEM levels. These are the necessary commands:

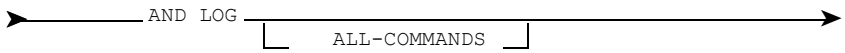
```
CREATE dict MASTER 'password'
INDEX xxx SOURCE xxx DATA xxx ;
DICTIONARY dict ;
AUTHORITY 'password' ;
RESTORE SOURCE FROM-STATUS DMR NO-PRINT ;
RESTORE SOURCE FROM-STATUS name REPLACE NO-PRINT ;
```

where *name* = CONV or UDS

```
CONTROL NEW-ALIASES ;
CONTROL UDS DU012 ;     (Include only if status UDS is restored)
BULK ENCODE ;
```

The CONTROL NEW-ALIASES command is described in Chapter 12, "Commands for Dictionary Controllers," on page 87. The current version of the command supersedes use of the DALIAS macro supplied in Manager Products releases prior to ControlManager Version 2 release 2.1.

If the DEMO dictionary is to be used as a training aid and the logging facility is to be used to monitor students' progress, then this clause can be added to the CREATE commands given above:



ALL-COMMANDS is only available if the Audit and Security facility is installed.

Restoring the User Interface Dictionary

For those implementations of ControlManager that include the User Interface facility and/or the User Defined Output facility, a dataset, MP.UIDICT, is included on the installation tape. This dataset contains a dictionary of the User Interface output record formats and Access Call control parameter area definitions. These members include information relating to User Defined Output parameter numbers.

MP.UIDICT is the output of a SAVE ALL command issued on the Manager Products User Interface dictionary. The Controller can retrieve this dictionary by a RESTORE ALL command. An example job control to do this can be found in the Manager Products installation manual appropriate to your environment.

The dictionary into which the User Interface dictionary is restored should have been created by this command sequence:

```
CREATE dict MASTER 'password'
INDEX xxx SOURCE xxx DATA xxx
WITH n STATUSES ;
```

where *n* is defined in the appropriate User Interface Facility publication.

The approximate number of members contained in MP.UIDICT is published at the same time as the number of statuses.

7 Control of Member Locking

This chapter includes these sections:

Introduction to Member Locking for Controllers

The member locking feature is provided by the ControlManager Workstation Interface Tailoring facility (selectable unit CMR-WS01) and is used to prevent concurrent updating of dictionary members.

All users can lock both new and existing dictionary members, and release their own locks, but you the Controller are responsible for controlling member locking and lock information, as detailed:

- Specifying the default lock period for a dictionary. The lock period is the length of time after which a lock expires. The lock period commences as soon as the LOCK command is entered.
- Specifying lock periods for individual users.
- Monitoring which members have been locked by each user. You can list these locked members, when you display users' security record details with the SECURITY LIST GIVING LOCKS command.
- If necessary, releasing members locked by any other user in the dictionary, using the LOCK RELEASE command.

Manipulating Locked Members

You can limit the operation of the ANALYZE and SAVE commands in order to process locked members only, by including the LOCKED keyword in the command.

For example, to analyze the block usage details of locked members only, enter:

```
ANALYZE LOCKED GIVING DETAILS ;
```

When you reorganize and copy a dictionary with the SAVE ALL and RESTORE ALL commands, all lock details from the old dictionary will be carried over into the receiving dictionary.

A RESTORE SOURCES command (entered following a SAVE command) restores only members' source records. However, a source member from the original dictionary cannot be copied into the receiving dictionary if a member of the same name is currently locked by any user other than the Controller in that receiving dictionary.

All variants of the LOCK command are updating commands. Therefore, if either the ALL-COMMANDS or UPDATES clauses are included in any of these commands, any LOCK commands entered during the current PRODUCTs session will be included in the resulting output:

AUDIT	CREATE
LOG	RELOAD

Commands Inhibited by Member Locking

These commands either cannot be used on a member which is locked by another user, or do not register any lock information when they are processed:

CONSTRUCT UDS-TABLE	COPY
STATUS FREEZE	STATUS MERGE
STATUS PURGE-DATA-ENTRIES	STATUS REMOVE
STATUS UNFREEZE	

Refer to Chapter 12, "Commands for Dictionary Controllers," on page 87 for details of these commands.

For example, if, at the time you enter the STATUS FREEZE command (to change a status named DEV1 from an update status to read-only status), there are six members currently locked in that status, the locks on those members will not be reflected when DEV1 becomes a read-only status.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for the commands available to all users which are inhibited by member locking.

8 Status Facilities for Controllers

This chapter includes these sections:

Status and the Dictionary Controller

As dictionary Controller you are responsible for these specific status-related tasks:

- Creating statuses in the dictionary
- Establishing the status structure of the dictionary by defining the relationship between its statuses
- Maintaining the status structure

Your primary responsibility is to utilize the status facility in the way best suited to the needs of your organization. This chapter describes the concepts and commands you need to understand to be able do this.

Before reading this chapter you should be familiar with the basic concepts of status as well as the unrestricted documentation available to all users.

To find out which status facility your organization has installed, enter:

```
ENVIRONMENT ALL ;
```

This will produce a list of all installed Manager Products and selectable units. If only Basic Status is listed then that is the facility you are using. If both Basic and Advanced Status are listed then you are using Advanced Status.

Refer to these publications for unrestricted details of the Status facilities:

- *ASG-Manager Products Status Concepts*
- *ASG-Manager Products Basic Status*
- *ASG-Manager Products Advanced Status*

Establishing Statuses in a Dictionary

Establishing statuses in the dictionary is a two-step operation:

1 When the dictionary is created it must be created with a specified number of unnamed statuses. This is done using the WITH clause in the CREATE command.

2 Unnamed statuses must be named when they are required for use in the dictionary, their disposition designated as either read-only or update, and their relationships with other named statuses defined.

So, your dictionary consists of named statuses and unnamed statuses. Unnamed statuses are available statuses that are not in use. Named statuses are statuses which are currently in use or which have been used. To find the total number of each, enter:

```
STATUS LIST DICTIONARY ;
```

Unnamed statuses are named using the STATUS NAME command and named statuses may be renamed using the STATUS RENAME command.

Members added to the dictionary before a status is named are automatically assigned to the first status named subsequently.

Considerations When Creating a Dictionary with Statuses

For statuses to be established in the dictionary, the permissible number of statuses for that dictionary must be specified when the Controller creates it. This is done using the WITH clause in the CREATE/VCREATE command. The maximum number that may be specified is 255, the minimum is 1.

Dictionary creation affords you the only opportunity to specify the number of statuses that the dictionary may contain. In order to increase or decrease that number you must create a new dictionary with the new number of statuses, and incorporate the old dictionary into it (using the SAVE and RESTORE commands).

The CREATE/VCREATE command establishes the specified number of statuses in the dictionary as unnamed statuses. These must be named when they are required for use in the dictionary. Unnamed statuses are not usable or visible to users of the dictionary, but they do take up disk space and processing time.

Before you specify the number of statuses with which the dictionary is to be created, you must consider these things:

- The cost in disk space and processing time of overestimating the number of statuses that you are likely to require
- The cost and time involved in creating a new dictionary if you underestimate

Building and Maintaining Status Structures

You will be using the same commands to build a status structure in a new dictionary as to amend an existing status structure. These commands are variants of the STATUS command:

```
STATUS NAME                STATUS PERMIT
STATUS RENAME              STATUS MERGE
STATUS FREEZE              STATUS PURGE-DATA-ENTRIES
STATUS UNFREEZE            STATUS REMOVE
                           STATUS DEFAULT
```

These commands are common to the Basic Status and the Advanced Status facilities. The status structures you can achieve by using them differ to the extent that with Advanced Status you can designate base statuses as updatable statuses. With Basic Status all base statuses are read-only by default and may not be changed to update.

You establish unnamed statuses in the dictionary when you CREATE it. To bring any of those statuses into active use you must name them using the STATUS NAME command. They may be renamed subsequently using STATUS RENAME. When a dictionary has unnamed statuses but no named statuses or has only one named status, it behaves exactly the same as a dictionary without a Status facility, except that processing time may be affected. A newly named status will be a non-base status by default.

The STATUS FREEZE command is used to establish base statuses and as such is the primary command with which to establish the structure of the statuses in your dictionary. The effect of freezing a status is that it becomes the direct base status of all non-base statuses in the dictionary and, if it is not the first status to be frozen, the direct dependent of the status frozen before it. Thus a status hierarchy is assembled, initially, using the STATUS NAME and STATUS FREEZE commands.

A STATUS UNFREEZE command reverses the effect of STATUS FREEZE by moving a base status to a non-base status.

Non-base statuses created by a STATUS NAME or STATUS UNFREEZE command default to update statuses. Base statuses default to read-only statuses.

The STATUS PERMIT command may be used to change the read-only or update condition of a status. However, if you are using Basic Status you cannot use it to make a base status updatable.

The STATUS DEFAULT command enables you to designate the status into which users accessing the dictionary are taken automatically.

STATUS REMOVE and STATUS PURGE-DATA-ENTRIES enable you to remove a status entirely and to remove a status's data entries records, respectively.

The STATUS MERGE command enables you to combine two statuses into one, providing that the content of those statuses meet certain conditions.

A detailed description of each command and its use is given in Chapter 12, "Commands for Dictionary Controllers," on page 87.

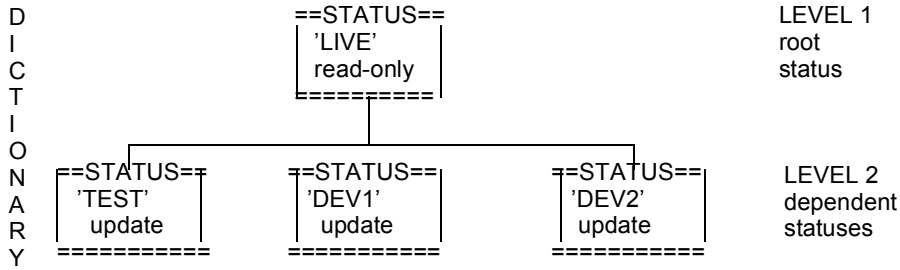
Example: Setting Up a Status Structure

Assuming that the dictionary has been created with at least 5 statuses, these commands:

```
STATUS NAME LIVE, TEST, DEV1, DEV2;
STATUS FREEZE LIVE ;
STATUS DEFAULT LIVE ;
```

would create the status structure represented in Figure 3:

Figure 3 • Status Structure



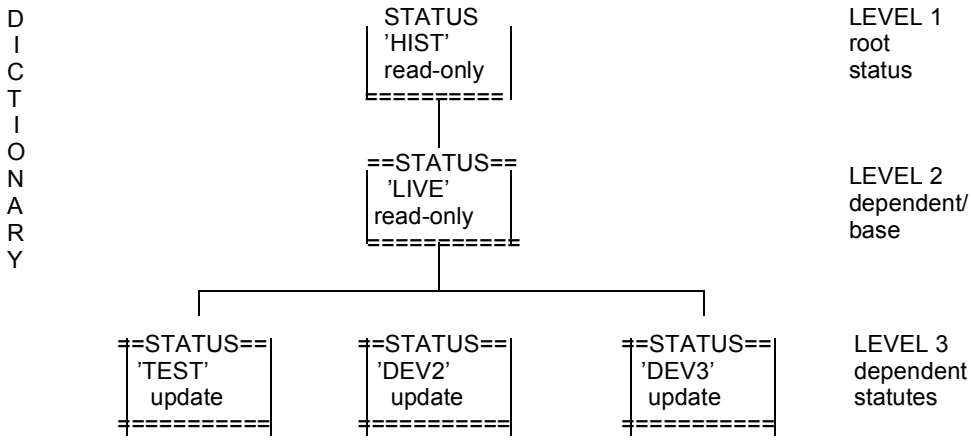
Status LIVE contains documentation of the implemented system. DEV1 and DEV2 are used by different teams involved in developing different aspects of that system. TEST is used by both teams.

If the development project being documented in DEV1 is implemented, then for DEV2 and TEST to be based upon the status which reflects the implemented system, it will be necessary to base them on DEV1, to rename the statuses appropriately, and perhaps to name another status for ongoing development. This is accomplished with these commands:

```
STATUS FREEZE DEV1 ;
STATUS RENAME LIVE AS HIST, DEV1 AS LIVE ;
STATUS NAME DEV3 ;
STATUS DEFAULT LIVE ;
```

The resulting status hierarchy is represented by Figure 4:

Figure 4 • Status hierarchy



If there is no requirement to retain HIST then it can be removed using the STATUS REMOVE command. When applied to a root status, STATUS REMOVE actually merges that status into its direct dependent status so that definitions for members that do not exist in the dependent are not lost from the dictionary.

Merging a Root Status with Its Direct Dependent Status

When you REMOVE a root status, the definitions it contains are not actually removed—they are merged into the root status's direct dependent status (which must itself be a base status for the STATUS REMOVE command to be accepted).

The reason for this is to maintain the dependency of the direct dependent status and the dependency of any indirect dependent on the root status. The root status may contain definitions of members which have not been redefined in the dependent statuses but are visible from them.

If such members were simply removed with the root status, they would also be effectively removed from each dependent status from which they were visible.

If every member recorded in the root status also has a record in its direct base status, then every definition in the root status is effectively removed.

Merging Sibling Statuses

If you attempt to merge sibling statuses (that is, non-base statuses that have the same direct base status) which are diverging, this error message will be output:

```
OVERLAP BETWEEN NON-BASE STATUSES
```

The command is rejected and processing continues with the next command. Statuses are diverging if they are sibling statuses and if these conditions are present:

- Any member has a source record in both of the statuses
- Any member has a data entries record in both statuses
- Any index-name (catalog classifications, aliases, and user-defined indexed attributes) is present in both statuses

A member has a data entries record in a given status if these conditions are present:

- The member has been encoded in that status (whether by UPDATE and FILE, RFILE, ENCODE, ADD, BULK ENCODE, MODIFY, or REPLACE commands)
- A dummy record has been created for the member in that status
- The usage of the member in that status differs from its usage in the most recently frozen of the earlier frozen statuses, if any. A difference in usage can be
  - — Use by a different set of directly referring members
  - — Use by the same set of directly referring members, but with at least one of the direct references being from a different clause of the referring member's data definition
- A member to which the member refers directly differs in member type between that status and the most recently frozen of the earlier frozen statuses

You may be able to remove a member’s source record and associated data entries record using a REMOVE or a REVERT command. The REINSTATE command enables you to remove unencoded source records.

The STATUS PURGE-DATA-ENTRIES command enables you to remove all data entries records for a specified status.

Interrogating the Dictionary for Diverging Statuses

If you are using Basic Status, you can enter this command, in either of the statuses to be merged, in order to produce output which will assist you in identifying diverging statuses:

```
LIST ALL-STATUSES CURRENT INDEX-NAMES ;
```

If there is a significant difference in the size of the two statuses, enter the command in the smaller one, so the command will process faster.

With Advanced Status you can obtain more specific information about divergence between statuses. Adjust your Status Window so that the Window Range for Selection, Output, and Diverging includes only the two statuses you wish to merge. Then enter:

```
LIST ALL-STATUSES DIVERGING INDEX-NAMES ;
```

This will produce a list of the members whose definitions diverge between the two statuses to be merged.

Restartable STATUS FREEZE and STATUS UNFREEZE

STATUS FREEZE and STATUS UNFREEZE are restartable commands for BDAM or VSAM-organized dictionaries only. You cannot restart the commands where a DIV-organized dictionary is used. This means that if, for any reason, the execution of a FREEZE or an UNFREEZE fails before it is complete, you can restart the procedure without necessarily having to run it again from the beginning. Instead, you can restart the FREEZE or UNFREEZE from the point where the failure occurred.

This section explains how the STATUS FREEZE and STATUS UNFREEZE operations are carried out and how to use the restart facility.

During the operation of a STATUS FREEZE or STATUS UNFREEZE, multiple members are re-encoded in many statuses. This stage is called a mass re-encode stage, and it occurs in several passes.

When a pass is ended, this message is output:

```
DM112961 MASS RE-ENCODING PASS n ENDED AT hh.mm.ss
```

During the mass re-encode stage, the dictionary is automatically disabled and the automatic error recovery system is switched on.

Before the start of the first pass of the mass re-encode, and after each two percent of the members have been encoded, a checkpoint is written. A checkpoint contains the information necessary for a subsequent restart, that is, it acts as a marker which indicates that all the members up to that point have been re-encoded.

Before the mass re-encode stage starts, these messages are output:

```
DM11299I         CHECKPOINT WRITTEN
DM11294I         MASS RE-ENCODING PASS n STARTED AT hh.mm.ss
                                        RUN IS RESTARTABLE
```

After each subsequent checkpoint has been written, these messages are output:

```
DM11299I             CHECKPOINT WRITTEN
DM11295I             MASS RE-ENCODING. nnnn MEMBERS PROCESSED
                                              BY hh.mm.ss
```

If, during this stage, the FREEZE or UNFREEZE fails, you can restart the run from the last checkpoint by entering the appropriate command:

```
STATUS FREEZE ;
```

or

```
STATUS UNFREEZE ;
```

To start the FREEZE or UNFREEZE from the beginning again, enter, as appropriate:

```
STATUS FREEZE status-name ;
```

or

```
STATUS UNFREEZE status-name ;
```

On restart, these messages are output:

```
DM11297I  MASS RE-ENCODING RESTARTED - RUN IS RESTARTABLE
DM11299I  CHECKPOINT WRITTEN
DM11294I  MASS RE-ENCODING PASS n STARTED AT hh.mm.ss -
                                        RUN IS RESTARTABLE
DM11298I  RESTART MEMBER member-name REACHED
```

The member named is the first one which will be processed in the restart run. The process will continue in the same way as before the interruption.

You can use the restart facility any number of times during the execution of a STATUS FREEZE or a STATUS UNFREEZE procedure, until it is complete. The procedure is cumulative and the only members not processed, in the event of a failure, are those which occur after the writing of the last checkpoint.

Security by Status

Different statuses in your dictionary may be used by different departments, project areas or even by different organizations. In this context it may be necessary to protect the information contained in each status from unauthorized access and/or to provide different types of access to different classes of user.

The level of security by status that you can achieve is dependent upon the Manager Products selectable units you have installed. This section summarizes what can be done with different configurations of selectable units.

The Audit and Security facility (selectable unit CMR-DD3) enables you to protect dictionary members against access by unauthorized users. However, the protection of a member will be effective for all definitions of that member, in all statuses.

The Status facilities themselves provide elementary security features:

- Statuses can be read-only or update
- The STATUS command can be disabled for dictionary users by placing this command into a Global Profile:

  ```
  SET PRIMARY COMMAND STATUS OFF ;
  ```

  Users are then restricted to working in the default status.

Advanced Status also provides a Status Window capability with restricted (Controllers only) capabilities that enable you to limit the statuses to which a user has access for each of a number of activities. For example, you can designate the statuses that a user may not update and the designation will override the read-only/update condition of the status established either as a default, when it was named, or by use of the STATUS PERMIT command. However, to take advantage of these capabilities, for the purpose of implementing security by status, the Systems Administrators Environmental Control Facility must be installed.

If the Systems Administrators Environmental Control Facility (SAECF) is installed, the Manager Products environment can be tailored for individual users through Logon Profiles. With this capability you can disallow the STATUS command for individual users and, by placing a STATUS *status-name* command into their Logon Profiles, provide automatic access after Logon to the required status (provided that the Profiles also provide access to a dictionary).

SAECF also provides the ability to disallow any updates to the dictionary from individual users. This would enable you to allow a particular user read-only access to all statuses in the dictionary.

Options for inclusion in a Logon Profile depend on which of the Status facilities you are using and are described in separate panels.

A more flexible approach to implementing security by status is possible when both the SAECF and the Procedures Language facility are installed. Corporate Executive Routines may be used in place of, or in addition to, Logon Profiles, to define a dictionary and/or a discrete status environment for individual and/or groups of users. Each Corporate Executive Routine can be restricted to use by particular users/groups of users and/or executed via a Logon Profile when a user logs on to Manager Products.

Options for inclusion in Corporate Executive Routines depend on which of the Status facilities you are using and are described in separate panels.

Since Global Profiles, Logon Profiles, and Corporate Executive Routines are controlled by the Systems Administrator, you must meet with that person to arrange the Profiling and Corporate Executives you require in order to use many of the methods described above.

The Procedures Language includes conditional logic capabilities. These can be used to implement access control to Corporate Executive Routines by testing variables such as Logon ID and user class as well as various system variables.

Note:

Consideration must be given to the security system you have implemented, or want to implement, when you use any of the commands that change and/or remove status names from the dictionary. Changing or removing a status name may necessitate your updating many Profiles and/or Corporate Executive Routines in order to maintain that system.

Activities Controlled by Maximum and Minimum Window Range

This table indicates the activity controlled by each Window Type when a MAXIMUM or MINIMUM Window Range has been specified.

Window Type	Related Activity
SELECTION	Statuses from which members are selected by:
	ALL-STATUSES keyword
	LIST HISTORY command
	IF clause in time-and-user-related-selection.
OUTPUT	Statuses from which details of members selected by:
	ALL-STATUSES keyword
	LIST HISTORY command
	IF clauses in time-and-user-related-selection
	COPY command specifying FROM are output
DIVERGING	Statuses searched for diverging member definitions.
READ-ONLY	Statuses available for read-only access.
STATUS-LIST	Statuses output in response to a STATUS LIST command.
UPDATE	Statuses available for update access.

As supplied by ASG, the Status Window has a default maximum and minimum Window Range of DICTIONARY for all Window Types.

Implementing Security by Status—Basic Status and SAECF

The SAECF provides the ability to tailor an individual user's environment through the Logon Profile. It additionally provides a command which, when placed in a Logon Profile, will prevent a user from updating the dictionary.

So, in addition to designating statuses as read-only or update, you can prevent a user from updating in all update statuses by placing this command into that person's Logon Profile:

```
SET DICTIONARY-UPDATES OFF ;
```

This will prevent the user from updating in all statuses (in all dictionaries) without preventing read-only access.

Placing the appropriate DICTIONARY, AUTHORITY, and STATUS *status-name* commands into a user's Logon Profile provides automatic access to the specified dictionary and status when that user logs on to Manager Products.

Basic Status, SAECF, and UDC

Facilities

The User Defined Commands facility (UDC) enables the Systems Administrator to set up Corporate Executive Routines. These are stored sets of commands that are executed when a user enters the name of the Routine. They may include commands normally restricted to dictionary Controllers or Systems Administrators.

If the Systems Administrators Environmental Control Facility (SAECF) is also installed, an Access Control Level may be specified for each Executive so that it may be executed only by users that have the required Executive Level or Executive Group Level specified in their Logon Profile. This combination of features enables you to achieve this security by status:

- Access to dictionaries may be tailored for individual users.
- Access to statuses within each dictionary may be tailored by individual users.
- Individual user's ability to update a dictionary and/or a status may be tailored so that one user may update in a particular dictionary/status and another user is prevented from doing so.

Implementation

Access to statuses may be controlled via Corporate Executive Routines and Logon Profiles as explained in this section.

Users' access to dictionaries and/or statuses can be limited through Logon and/or Global Profiles as follows:

- If there are several dictionaries at your installation, prevent users accessing all dictionaries by including this command:

  ```
  SET PRIMARY-COMAND DICTIONARY OFF ;
  ```

  and/or

- Prevent individual and/or groups of users from accessing any status, through the STATUS command, by including this command:

  ```
  SET PRIMARY-COMMAND STATUS OFF ;
  ```

  Note:

  The disadvantage with this option is that you must then create a Corporate Executive Routine for each STATUS command variant, such as STATUS LIST, that will still be required for use.

- Prevent users updating in any status, including update statuses, by including this command:

  ```
  SET DICTIONARY-UPDATES OFF ;
  ```

- Include the appropriate DICTIONARY, AUTHORITY, STATUS (if the dictionaries default status is not suitable), and SET commands in order to provide automatic access to the required dictionary/status environment when a user logs on to Manager Products.

Having limited access to dictionaries and/or statuses through Logon and/or Global Profiles, you can provide controlled access through Corporate Executive Routines with these actions:

- Set up a Corporate Executive Routine for each discrete dictionary and/or status environment required. Include in each a selection of these commands as appropriate in relation to the commands indicated in Logon and Global Profiles:

  ```
  DICTIONARY dictionary-name
  AUTHORITY password
  STATUS status-name
  SET DICTIONARY-UPDATES ON/OFF
  ```

- Specify an Access Control Level for each Corporate Executive such that only the required users will be able to execute it and thus invoke the dictionary/ status environment to which it gives access.

Note:

If the same status name is included in several dictionaries, a Corporate Executive Routine that includes a STATUS *status-name* command specifying that status-name, and does not also include a DICTIONARY and an AUTHORITY command, will provide access to that status in whichever dictionary is current.

Advanced Status, SAECF, and UDC

Facilities

The features available when the above combination of selectable units is installed incorporate those available when the SAECF and the UDC facility are installed with Basic Status.

In addition, the STATUS WINDOW command is available which, when placed in Logon Profiles and/or Corporate Executive Routines, can be used to control these things:

- The statuses to which a user has update access
- The statuses to which a user has read-only access
- The output from STATUS LIST command
- The range of statuses available to each user for these activities:
  - Using the ALL-STATUSES keyword (in LIST, PRINT, REPORT, and GLOSSARY commands)
  - Using the LIST HISTORY command
  - Using the status-related-selection keyword DIVERGING
  - Using the IF clause in time-and-user-related-selection
  - Using the FROM clause in the COPY command

If SAECF is installed, this control is accomplished by placing the appropriate commands into a user's Logon Profile.

If UDC is also installed, then Corporate Executive Routines, which include the appropriate commands, can be used to control access to dictionaries and/or the statuses they comprise.

Several Executive Routines may be used to define several different dictionary and/or status environments and access to those environments controlled by restricting use of the Executive Routines to the required users. The requirement may be to cater to different development teams, departments, and/or different classes of dictionary user.

Thus it is possible to tailor the status environment for individual and/or groups of users so that the required statuses are secure from unauthorized access.

Implementation

Access to dictionaries and/or statuses may be controlled through Logon Profiles, if the SAECF is installed, and/or Corporate Executive Routines, if the UDC is also installed. Using both facilities you can provide a default status environment for each user and/or alternative environments relating to different tasks, projects, or user class.

Using Profiles

In Logon and/or Global Profiles:

- Include a STATUS *status-name* command so that the required status is accessed automatically when the dictionary is accessed (if the dictionaries default status is not suitable).
- Include a STATUS WINDOW MAXIMUM/MINIMUM command using the options necessary to create the required status environment. The DICTIONARY command causes the Status Window Maximum and Minimum Window Ranges to revert to their unrestrictive default, therefore you must also include a SET PRIMARY-COMMAND DICTIONARY OFF command to prevent its use. If there are several dictionaries at your installation, then it may be necessary to provide access both to the dictionaries and to the statuses they contain, via Corporate Executive Routines.

Note:

You do not need to disable the STATUS command in order to prevent users accessing a status; you can achieve the same effect by excluding statuses that you do not want a user to access from that user's Status Window Maximum for update and for read-only.

Using Profiles and Corporate Executives

Include in Logon and/or Global Profiles:

- Any of the options described above
- A SET PRIMARY-COMMAND DICTIONARY OFF command
- The name of a Corporate Executive Routine which provides access to the required dictionary and/or status environment

Create Corporate Executive Routines this way:

- Construct a separate Corporate Executive Routine for each dictionary and each discrete status environment you wish to create over and above the environments accessed through Logon and Global Profiles.
- Include in each the appropriate combination of these commands:

```
DICTIONARY
AUTHORITY
STATUS status-name
STATUS WINDOW MAXIMUM/MINIMUM
SET PRIMARY-COMMAND STATUS ON
SET PRIMARY-COMMAND DICTIONARY ON
```

- Specify an Access Control Level for each Corporate Executive such that only the required users will be able to execute it and thus invoke the dictionary and/or status environment to which it gives access.

Note:

If the same status name is included in several dictionaries, a Corporate Executive Routine that includes a STATUS *status-name* command specifying that *status-name* and does not also include a DICTIONARY and an AUTHORITY command will provide access to that status in whichever dictionary is current.

The STATUS PERMIT command is not suitable for use in Corporate Executive Routines or Logon Profiles when the intention is to provide different access for individuals/groups.

Basic Status: Profiles and Corporate Executive Routines

These are examples of Logon Profiles:

Example 1

```
LOGON-PROFILE
PASSWORD GERRY
EXECUTIVE-GROUP 10
CONTENTS
SET PRIMARY COMMAND DICTIONARY OFF ;
SET PRIMARY-COMMAND STATUS OFF ;
```

Example 2

```
EXECUTIVE-ROUTINE
EXECUTIVE-LEVEL 10
DESCRIPTION 'Access to Project2 Documentation'
DICTIONARY MIDWK ;
AUTHORITY TEAM1 ;
STATUS DEV2 ;
```

A user logging on and invoking the Logon Profile displayed in Example 1 accesses the dictionary MIDWK and status DEV2 by entering the name of the Executive Routine shown in Example 2.

These are examples of Corporate Executive Routines:

Example 3

```
LOGON-PROFILE
PASSWORD GERRY
EXECUTIVE-LEVEL 250
CONTENTS
SET PRIMARY-COMMAND DICTIONARY OFF ;
SET PRIMARY-COMMAND STATUS OFF ;
```

Example 4

```
EXECUTIVE ROUTINE
DESCRIPTION 'Access for Team Leader Project2'
EXECUTIVE-LEVEL 200
CONTENTS
SET DICTIONARY-UPDATES ON ;
DICTIONARY MIDWK ;
AUTHORITY MASTER ;
SET PRIMARY-COMMAND STATUS ON ;
STATUS DEV1 ;
```

A user logging on and invoking the Logon Profile shown in Example 3 can execute the Corporate Executive Routine described in Example 4 and subsequently access any status in the dictionary MIDWK.

Advanced Status: Profiles and Corporate Executive Routines

These are examples of Logon Profiles:

Example 1

```
LOGON-PROFILE
PASSWORD GERRY
EXECUTIVE-LEVEL 10
CONTENTS
SET PRIMARY-COMMAND DICTIONARY OFF ;
SET PRIMARY-COMMAND STATUS OFF ;
```

Example 2

```
EXECUTIVE ROUTINE
DESCRIPTION 'Access to project2 documentation'
EXECUTIVE-LEVEL 10
CONTENTS
DICTIONARY MIDWK ;
AUTHORITY TEAM1 ;
STATUS DEV2 ;
STATUS WINDOW MAXIMUM FOR SELECTION IS ONLY STATUSES DEV1,
DEV2, LIVE, HIST
FOR OUTPUT IS DICTIONARY
FOR STATUS-LIST IS ONLY STATUSES DEV1, DEV2, LIVE, HIST
FOR UPDATE IS CURRENT
FOR READ-ONLY IS ONLY STATUSES DEV1, DEV2, LIVE, HIST;
```

A user logging on and invoking the Logon Profile detailed in Example 1 accesses the dictionary MIDWK and the status DEV2 by entering the name of the Executive Routine displayed in Example 2. The STATUS WINDOW MAXIMUM command defines the status environment in which the user will work.

These are examples of Corporate Executive Routines:

Example 3

```
LOGON-PROFILE
PASSWORD ANDY
EXECUTIVE-LEVEL 250
IDENTITY AA1
CONTENTS
SET PRIMARY-COMMAND DICTIONARY OFF ;
SET PRIMARY-COMMAND STATUS OFF ;
```

Example 4

```
EXECUTIVE ROUTINE
DESCRIPTION 'Access for team-leader project2'
EXECUTIVE-LEVEL 200
CONTENTS
DICTIONARY MIDWK UPDATE ;
SET PRIMARY-COMMAND STATUS ON ;
STATUS DEV2 ;
```

A user logging on and invoking the Logon Profile shown in Example 3 accesses the dictionary MIDWK by entering the name of the Executive Routine which appears in Example 4. Thereafter, that user is free to define the status environment and may use any STATUS command variant, except those restricted to dictionary Controllers. However, access to other dictionaries would require execution of another Corporate Executive Routine.

Documenting Partial or Phased Implementation

Partial Implementation—Scenario

Under certain circumstances you may want to move part of the contents of a dependent status, typically a status used to document development of a system, into its direct base status, typically a status that documents the implemented system. For example, time and/or other business constraints may have necessitated partial implementation of a development project and the dictionary is required to reflect the movement of completed work from the development to the production environment.

In these circumstances, the initial difficulty is to identity those definitions that reflect completed development work from the other definitions in the status used to record the development work. This difficulty may be alleviated if this procedure is adopted:

- As new and changed definitions are added to the development project's status in the data dictionary, an entry is made in the CATALOG clause to indicate that the definition is under development. The entry might be TENTATIVE, for example, or DEVELOPMENT.
- When a phase or stage of the development cycle is completed, appropriate members' catalog entries can be modified. The entries might be APPROVED, for example, or IMPLEMENTed.

If it is decided to put the development project into production in phases, adopting this cataloging method will enable you to manipulate those definitions within the development status that relate to completed development work.

The procedure for moving selected definitions from a dependent status into a base status differs according to which status facility you are using. If you are using Basic Status then all base statuses are read-only. Consequently, the procedure involves naming a new status, copying the required members into that status, then merging it into the most recently frozen status or freezing the new one. If you are using Advanced Status, base statuses may be updatable, and it is a simple matter to copy the required definitions.

The REVERT command available with both facilities enables you to remove the definitions from the dependent status once they have been successfully encoded in the base status.

Moving Members into a Base Status Using Basic Status

This section describes the recommended procedure for moving members from a dependent status into a base status.

Optionally use the UNLOAD command to back up the dictionary in case you make an error.

Name a new status (using the STATUS NAME command) into which the required definitions can be moved unless an empty non-base status is already available.

Keep the members that you want to move, for example:

```
STATUS dependent-status ;
KEEP AND LIST CURRENT DEFINITIONS KEPT ;
```

Copy the members contained in the KEPT-DATA list into the new status, for example:

```
STATUS new-status ;
PERFORM
"COPY '*' FROM dependent status ENCODED-SOURCE REPLACE
                                        OLD-DATE" KEPT ;
```

Check that the members have been copied into the new status successfully then encode them:

```
BULK ENCODE UNVERIFIED KEPT ;
```

Note:

UNVERIFIED is useful as it may be necessary to repeat the BULK ENCODE command, because of member type changes and dependencies on those changes, until all the members are encoded successfully.

If and when all the members are encoded successfully, freeze the new status or merge the new status into the most recently frozen status.

Use the REVERT command to remove the definitions moved from the dependent status:

```
STATUS dependent-status ;
PERFORM 'REVERT *' KEPT ;
```

Optionally rename the dependent status using STATUS RENAME.

Optionally use the UNLOAD command to back up the dictionary again and thereby safeguard the changes made against loss by system failures or malfunctions. Your decision to do so will depend upon the size and/or number of the changes made and on whether ROLL-FORWARD is possible (this depends on whether Logging is implemented). Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of backing up your dictionary.

Note:

If the UDC facility is installed, an Executive Routine may be written to perform the above procedure for individual members.

Moving Members into a Base Status Using Advanced Status

This section describes the recommended procedure for moving members from a dependent status into a base status.

Optionally use the UNLOAD command to back up the dictionary in case you make an error.

Keep the members that you wish to move from the dependent status into the base status, for example:

```
STATUS dependent-status ;
KEEP AND LIST CURRENT DEFINITIONS MEMBERS name ;
```

Copy the members contained in the KEPT-DATA list into the base status:

```
STATUS PERMIT UPDATE base-status ;
STATUS base-status ;
PERFORM
"COPY '*' FROM dependent-status ENCODED-SOURCE REPLACE
                                          OLD-DATE" KEPT ;
```

Check that the members have been copied into the status successfully and then encode them:

```
BULK ENCODE UNVERIFIED KEPT ;
STATUS PERMIT READ-ONLY base-status ;
```

Note:

UNVERIFIED is useful as it may be necessary to repeat the BULK ENCODE command, because of member type changes and dependents on those changes, until all the members are encoded successfully.

If and when all the members are encoded successfully, use the REVERT command to remove the moved definitions from the dependent status:

```
STATUS dependent-status ;
PERFORM `REVERT *' KEPT ;
```

Optionally rename the dependent status using STATUS RENAME.

Optionally use the UNLOAD command to back up the dictionary again and thereby safeguard the changes made from loss by system failures or malfunctions. Your decision to do so will depend upon the size and/or number of the changes made and on whether ROLL-FORWARD is possible (this depends on whether Logging is implemented).

Note:

If the UDC facility is installed an Executive Routine may be written to perform the above procedure for individual members.

Moving Members into a Base Status Using Executive Routines

If the UDC facility is installed (selectable unit CMR-UD05) you could create an Executive Routine (Corporate or User) to move a single member from a dependent status into its base status then REVERT the definition in the dependent status. Conditional logic can be used to ensure that the member is successfully copied and encoded in the base status before the definition in the dependent status is removed via REVERT. Figure 5 is an example of the contents of such an Executive.

Figure 5 • Create an Executive Routine

```
MPXX
STATUS base-status-name/new-status-name ;
COPY '&PO' FROM dependent-status-name ENCODED-SOURCE
                                        REPLACE OLD-DATE;
IF &CCOD EQ 0 THEN ENCODE '&PO'
IF &CCOD NE 0 THEN GOTO ERROR
STATUS dependent-status-name ;
REVERT '&PO' ;
EXIT
-ERROR
MESSAGE: WRITEL Error member not  successfulty encoded
EXIT
```

Note:

You could copy the above lines and use them to build the contents of an Executive—simply over-key the *base-status-name/new-status-name* and the *dependent-status-name* with real status names.

9 User Defined Syntax for Controllers

This chapter includes these sections:

Overview of User Defined Syntax

The User Defined Syntax Facility (UDS), selectable unit CMR-UD1, enables organizations to define the member types that they want to use in their dictionaries, together with the attributes required for those member types.

The User Defined Relationships (UDR) capability, which is a part of the UDS, enables organizations to define up to nine additional types of relationship between dictionary members, which can be used in addition to the pre-defined relationships.

If the UDS facility is included in your Manager Products configuration, then you, as Controller, are responsible for defining the member types that are to be available for use in your dictionary, together with their attributes and hierarchical relationships. The way you do this is summarized in this section.

Special member types:

- HIERARCHY
- MEMBER-TYPE
- MEMBER-TYPE-GROUP
- ATTRIBUTE-TYPE
- ATTRIBUTE-GROUP

are provided in the ASG-supplied UDS Table Dictionary. While recognizing that there will be exceptions, ASG recommends that each installation locate these member types in the Manager Products Administration Dictionary. The Systems Administrator (as Controller of the Manager Products Administration Dictionary) gives you the right to access and update that dictionary; that is, you become an authorized user of the Manager Products Administration Dictionary. The extent of your ability to access that dictionary may be restricted to particular statuses by your Logon Profile, or to particular members by the security levels and protections established in the dictionary, if the appropriate ControlManager selectable units are installed. Thus, your access rights as an authorized user may be restricted to the capabilities you require in order to set up the user-defined syntax for your own dictionary.

You are also responsible for defining the user-defined relationships which will apply to your dictionary.

See "Restoring ASG-supplied Dictionaries" on page 46 for the procedure for restoring the UDS Table Dictionary into your Manager Products Administration Dictionary. The job control requirements are given in the Manager Products installation manual appropriate to your installation.

Preparing Your Member Type Structure

Having first decided on the member type structure for your dictionary, you then enter the definitions of the member types and attributes into the Manager Products Administration Dictionary and encode them. You also add a HIERARCHY member, to specify which of the member type and attribute type definitions held in the dictionary are to form the UDS structure for your own dictionary. MEMBER-TYPE-GROUP and ATTRIBUTE-GROUP members can be used to simplify the overall development of your UDS structure. These member types are documented in the UDS branch of InfoBank.

This stage, the development of the hierarchical member-type structure for your dictionary and the specification of the required attributes, can be delegated to others. For you to delegate these tasks, the Systems Administrator will need to establish your nominated personnel as additional authorized users of the Manager Products Administration Dictionary; so delegation of the entering of UDS member definitions can only be done by mutual agreement between you, as a dictionary Controller, and the Systems Administrator.

The subsequent tasks cannot be delegated. They all require the presence of a keyword indicating Controller status in a Logon Profile before the relevant commands can be accepted. If you have this keyword in your Logon Profile you become a Designated Controller. While operating as a Designated Controller in a dictionary of which you are not the actual Controller, you are known as a Guest Controller.

Applying a UDS Table to Your Dictionary

When all your UDS member definitions have been entered into the Manager Products Administration Dictionary, you issue a CONSTRUCT UDS-TABLE command to construct a UDS-TABLE member in the MP-AID from your HIERARCHY dictionary member.

If your target dictionary is not a new one—that is, if it already contains members that were defined under a different UDS table—you will then need to issue a COMPARE UDS-TABLE command. This compares your new UDS table with the one previously in use for your dictionary, and outputs a UDS Table Comparison Report. This report warns you of any incompatibilities between your new structure and the old. If any such incompatibilities are present you must then decide what action to take to resolve them. The report tells you if you will need to re-encode the dictionary's members using the new UDS table.

The COMPARE UDS-TABLE command also sets up a UDS-COMPARISON-TABLE member in the MP-AID which can be subsequently accessed through an MP-AID LIST command.

When any incompatibilities have been resolved, you open your dictionary and issue a CONTROL UDS command. This establishes your new UDS table as the one required for your dictionary. It records in your dictionary the name of your UDS table; this is the MP-AID member name of your UDS-TABLE member.

The CONTROL UDS command also sets up a secondary UDS-TABLE member of the MP-AID. This indicates which table is in use for which dictionary. The primary UDS-TABLE member is the one established by your CONSTRUCT commands.

Commands are also provided for you to list UDS-TABLE and UDS-COMPARISON-TABLE members of the MP-AID.

The commands discussed above are specified in Chapter 12, "Commands for Dictionary Controllers," on page 87.

Implementing User Defined Relationships

The UDR capability provides nine user-defined relationships, which you can use to document relationships between members of any type available in a dictionary. These relationships are additional to the pre-defined relationships.

The UDRs can be tailored, to suit the requirements of each dictionary. The dictionary Controller is responsible for deciding whether to tailor them, and, if so, what the relationships are to be.

The default UDRs provided by ASG are UDR1 to UDR9. When you tailor UDRs in a dictionary (using the CONTROL UDR command) the default UDRs are renamed with the UDRs you specify.

It is also possible to remove UDRs (using the CONTROL UDR REMOVE command). This command restores the default UDRs.

When you define UDRs you should ensure that the definition is correct before dictionary members are defined with the UDRs you have specified. If you need to change UDRs you will have to be certain that no dictionary members have been defined with those UDRs, or, if they have, you will have to change them manually.

Similarly, when you remove UDRs from a dictionary, you will have to find out which members, if any, contain them, and delete them manually.

There is no automatic connection between the UDS table which applies to a dictionary and the UDRs which you define for that dictionary. This means that you have to implement UDRs independently of the implementation of the UDS table. If you apply a UDS table to a second or subsequent dictionary, you will have to implement UDRs separately in each of those dictionaries.

If you change or remove the UDS table which applies to a dictionary, the UDRs are not automatically changed or removed. You have to do that explicitly.

All nine UDRs are available for use with all the member types which are available in any dictionary, but you can restrict the use of UDRs in user defined members. You can do this by including in the definition of a user-defined member type, a RELATIONSHIPS VIA clause, as shown:

```
RELATIONSHIPS VIA UDR2 DISALLOWED ;
```

This clause will have the effect of disallowing the use of UDR2 (or its renamed keyword) in any members of the type being defined.

The version of the SHOW UDS command which is available only to Controllers, that is:

```
SHOW UDS TABLE table-name MEMBER-TYPES RELATIONSHIPS ;
```

shows the default UDRs (UDR1 to UDR9), in the output.

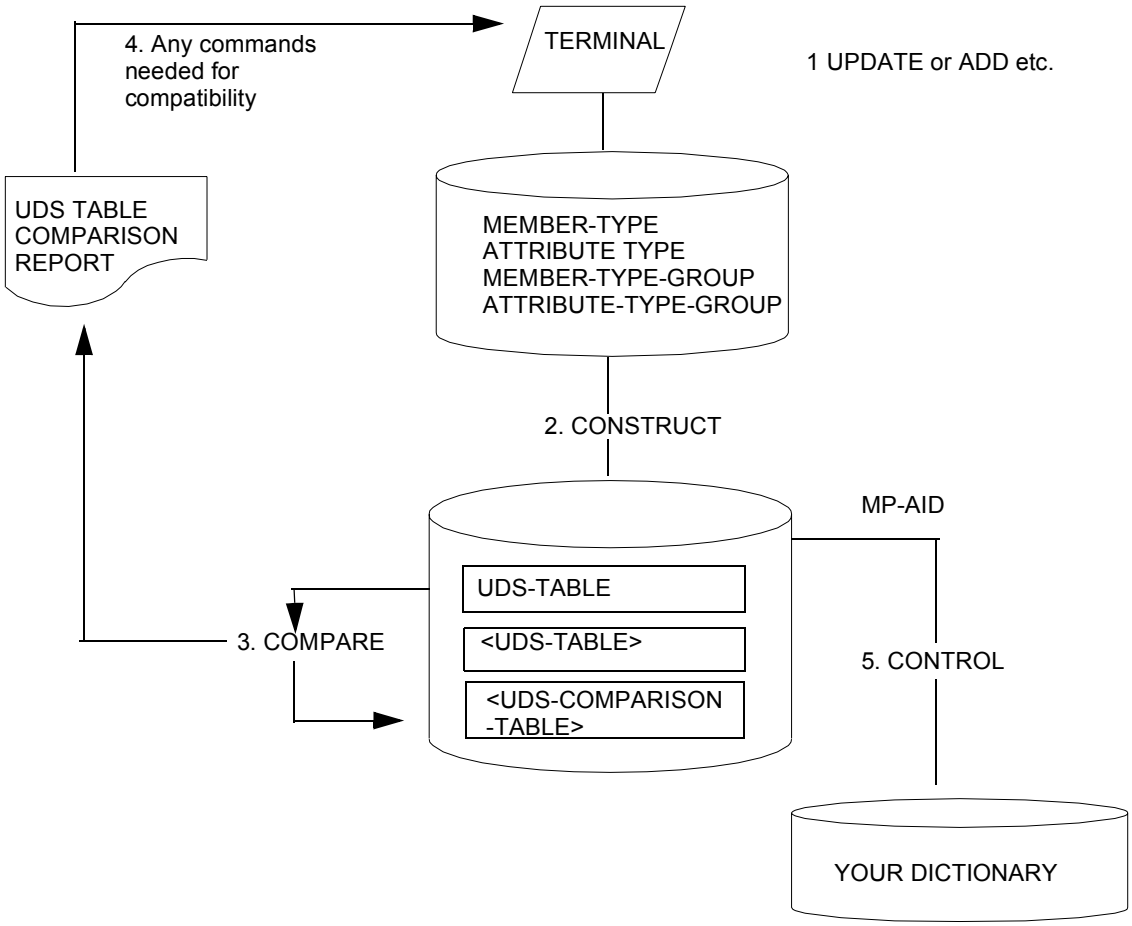
The version of the SHOW UDS command which is available to all users, that is:

```
SHOW UDS MEMBER-TYPES RELATIONSHIPS ;
```

which refers to the UDS table specific to the dictionary being accessed, shows the renamed UDRs in the output.

When you reorganize a dictionary using SAVE and RESTORE commands, renamed UDRs are retained.

Figure 6 • Implement User Defined Relationships



ASG-supplied UDS Tables

Supply and Applicability

A number of UDS tables are supplied by ASG as load modules. These, together with the MethodManager Information Engineering Knowledge Base (MIEKB) Structure constitute a preferred set of UDS tables which you can use in your dictionaries. If, when you issue a CONTROL UDS command (or a COMPARE UDS-TABLE command), ControlManager cannot find the table you want in the MP-AID, then it looks for it in the load library. The load module DU001 is used as a default UDS table when a dictionary is first CREATEd. (This applies whether or not the UDS facility is installed.)

The UDS table relevant to the Manager Products Administration Dictionary in UDS environments is DU007. If you set up your Manager Products Administration Dictionary following the recommended procedures, this UDS table will automatically be applied as part of the action of the RESTORE ALL of the UDS Table Dictionary. If you do not RESTORE ALL the UDS Table Dictionary into your Manager Products Administration Dictionary, you must CONTROL UDS DU007 with the latter dictionary open before you can encode any of the UDS member types (and hence before you can set up your own UDS tables). The UDS Table Dictionary contains the source members of all the preferred set UDS tables; if you RESTORE these members into your Manager Products Administration Dictionary as recommended, you can use them as a basis for deriving your own MEMBER-TYPE, ATTRIBUTE-TYPE, and HIERARCHY members, prior to CONSTRUCTing your UDS table.

The MIEKB Structure is not supplied as a source module. It can be constructed from the UDS Table Dictionary/Manager Products Administration Dictionary as UDS Table DU015. ASG recommends that you make use of the MIEKB Structure in your Corporate Dictionary/Repository, so that you are positioned to take advantage of the opportunities that will be offered by the future direction of Manager Products evolving technology.

The ASG-recommended procedures for setting up the Manager Products Administration Dictionary are described in Chapter 6, "Dictionary Reorganization, Copying, and Backup," on page 43.

"UDS Load Modules Supplied by ASG" on page 81 lists the names of UDS load modules supplied by ASG.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for information about the MIEKB structure.

UDS Load Modules Supplied by ASG

The following is a list of ASG-supplied load module names together with the member types which they support. These load modules constitute a preferred set of UDS tables which you can use for your dictionaries. If, when the dictionary Controller is comparing UDS tables or is applying a UDS table to a dictionary, a UDS table cannot be found on the MP-AID, then ControlManager looks for it in the load library.

Load Module Names	Member Types They Support
DU001	Default structure; supports all base member types excluding UDS member types
DU002	Supports all base member types except Manager Products Administration Dictionary member types; and supports the English language version of the Extended Data Processing Structure (EDPS)
DU003	Supports all base member types except Manager Products Administration Dictionary member types; and supports the Structured Analysis Structure (SAS)
DU004	Supports all base member types except Manager Products Administration Dictionary member types; and supports EDPS and SAS
DU006	Supports all base member types except Manager Products Administration Dictionary member types; and supports the Structured Development Structure (SDS)
DU007	Supports the Manager Products Administration Dictionary member types including UDS member types
DU012	Supports all base member types including the Manager Products Administration Dictionary member types including UDS member types; and supports EDPS
DUE02	Supports all base member types except Manager Products Administration Dictionary member types; and supports the Spanish language version of EDPS
DUF02	Supports all base member types except Manager Products Administration Dictionary member types; and supports the French language version of EDPS
DUG02	Supports all base member types except Manager Products Administration Dictionary member types; and supports the German language version of EDPS
DUH02	Supports all base member types except Manager Productss Administration Dictionary member types; and supports the Dutch language version of EDPS

Load Module Names	Member Types They Support
DUI02	Supports all base member types except Manager Products Administration Dictionary member types; and supports the Italian language version of EDPS
DUK02	Supports all base member types except Manager Products Administration Dictionary member types; and supports the Danish language version of EDPS
DUN02	Supports all base member types except Manager Products Administration Dictionary member types; and supports the Norwegian language version of EDPS
DUS02	Supports all base member types except Manager Products Administration Dictionary member types; and supports the Swedish language version of EDPS
DUU02	Supports all base member types except Manager Products Administration Dictionary member types; and supports the Finnish language version of EDPS

DU001 is used as a default UDS table when a dictionary is first created.

DU007 is intended for the Manager Products Administration Dictionary in UDS environments.

10 Miscellaneous Controller's Commands

In addition to the commands discussed under other headings, a number of miscellaneous commands are available to you as Controller. The capabilities offered by these commands, with the command identifier keywords, are listed here:

CONTROL CMS. To initiate or override concurrent access protection for a dictionary at the command level in CMS environments.

CONTROL ENQ-NAME. To specify the minor queue name (the *rname*) of the currently open dictionary.

CONTROL NEW-ALIASES. To incorporate a revised alias-type keyword list into the currently open dictionary.

CONTROL RESERVE. To control whether more than one central processor at a time can access the currently open BDAM dictionary.

DIAGNOSE. To check the integrity of the dictionary and report on any areas needing further attention.

DISABLE. To prevent the use of the dictionary by anyone other than the Controller and the Master Operator.

ENABLE. To cancel the effect of a DISABLE command; that is, to make the dictionary available again to other users.

XPRINT. To obtain selective printouts of internal formats of the dictionary, for ASG use in satisfying maintenance requests.

SET ENCODE-EXIT. To enable additional checking of dictionary members' sources during the ENCODE command, using the MPCMENCUX executive routine.

These commands also have Controller's extensions or operate differently when issued by the Controller of the currently open dictionary:

COPY. OLD-DATE keyword to copy to the new member the date information from the originating member.

DICTIONARY. NOLOG keyword to enable a dictionary to be opened without opening its log dataset

REMOVE. When issued by the dictionary Controller, can remove referenced members.

RESERVE Executive. SPEED keyword to maximize the speed of processing of Logical Units of Work.

For further information about any of the above-listed commands, please refer to Chapter 12, "Commands for Dictionary Controllers," on page 87.

11 The Master Operator

The Master Operator concept provides a secure mechanism for the Controller to delegate execution of certain of the private commands to the Operations area. The commands that can be delegated are those that are concerned with the physical backup and recovery of the dictionary. These commands can be delegated:

- DISABLE
- ENABLE
- JOURNAL
- LOG
- RELOAD
- ROLL-FORWARD
- UNLOAD

The mechanism for restricting the delegated execution of these commands to the designated Master Operator is a special password, set up separately for each dictionary by the Controller's version of the AUTHORITY command. The AUTHORITY command can also be used by the Controller to change the Master Operator's password whenever necessary in order to preserve security.

Once the Master Operator's password has been set up by the Controller, the Master Operator can issue AUTHORITY commands quoting that password. Any of the above-listed commands will subsequently be accepted. For the RELOAD and ROLL-FORWARD commands, a previous AUTHORITY command is not necessary. With these commands, the Master Operator's password is quoted within the command itself. The LOG STATUS and LOG ANALYSIS variants of the LOG command can also be issued as free-standing commands, with a dictionary name and Master Operator's password embedded. (If a dictionary is already open, however, this free-standing form is not accepted; a prior AUTHORITY command with an appropriate password is required.)

Acceptance by the Manager Product of the Master Operator's password does not give that operator access to the dictionary; any commands other than those listed above are rejected.

Since the Master Operator can neither interrogate nor (except to the extent entailed by the above-listed commands) update the dictionary, that person is not regarded as a user of the dictionary; the Master Operator is not included in the list of users output by a SECURITY LIST command. A Master Operator's Syntax Card is provided by ASG, giving the syntax of all dictionary management commands that can be used by the Master Operator. The Manager Products installation manual appropriate to your environment is also necessary for the Master Operator. This manual gives the job control and related requirements for running Manager Products. So that security can be preserved and control over the dictionary retained by the Controller, the Controller's Manual should not be made available to the Master Operator. Instead, specific job instructions should be issued to the Master Operator for the use of the available commands, in accordance with the installation's own standards.

12 Commands for Dictionary Controllers

This chapter includes these sections:

Command Descriptions

This section contains the descriptions of the commands or command variants whose use is restricted to dictionary Controllers. The description of each command is divided into three parts:

- A short description
- A detailed explanation
- Syntax

Some of these commands include secondary selection capabilities. The secondary selection capabilities are the same as those provided in certain Dictionary Management commands available to all users, and documented in *ASG-Manager Products Dictionary/Repository User's Guide*.

ANALYZE

The ANALYZE command is used to analyze the disk space utilization (that is, the logical block utilization) of all member/index-names in the dictionary. Use the SHOW UDS command to find out which member types are available in your dictionary.

Refer to "ANALYZE Syntax" on page 97 for the syntax of the ANALYZE command.

You can spell the command identifier either way: ANALYZE or ANALYSE. If you enter:

```
ANALYZE
```

without including any selections in the command, this is the assumed default:

```
ANALYZE INDEX-NAMES GIVING TOTALS ;
```

If you do not want this default analysis, you can refine the ANALYZE command in two ways:

- By the selection of particular members, using the selection and secondary selection clauses
- By the selection of the type of output required using the GIVING clause

If you include a selection in the ANALYZE command, you can select what you want to analyze:

- KEPT-DATA: members kept in a KEPT-DATA list
- LOCKED: currently locked members with encoded records
- MEMBERS *member-name-list*: a list of specific member names
- DUMMIES: dummy members
- INDEX-NAMES: types of member/index-names
- SOURCES: source records

Additionally, you can further refine the operation of the command by including a secondary selection in any variant of the ANALYZE command:

- By name
- By status
- By time
- By user

You can exclude particular types of member/index-names from being analyzed by including an EXCEPT clause in the command.

You can also specify how detailed you require the analysis output analysis to be, by using GIVING in the command.

Note:

Details relating to locking are available only if the Workstation Interface facility (selectable unit CMR-WS01) is installed.

Details relating to statuses are available only if either the Basic or Advanced Status facility (selectable units CMR-DD2 or CMR-AD21) are installed.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for a definition of member/index-names.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of member types and secondary selections.

Analyzing All Members, Source, Dummy, or Real Members

To analyze the disk space utilization of all members, enter:

```
ANALYZE ;
```

To analyze the disk space utilization of all members with a source record, enter:

```
ANALYZE SOURCES ;
```

Source members which have a dummy encoded record are also analyzed in this variant of the command.

To analyze the disk space utilization of all dummy members, enter:

```
ANALYZE DUMMIES ;
```

You can refine the ANALYZE DUMMIES variant of the command to exclude specific types of member/index-names from being analyzed by using EXCEPT.

For example, the command:

```
ANALYZE DUMMIES EQ EMP EXCEPT ITEMS ;
```

analyzes all dummy members which include the character string EMP in their names, excluding those member/index-names of the type ITEM.

You may use the keyword REAL to restrict analysis to non-dummy members, that is those which have definitions and are encoded.

For example, the command:

```
ANALYZE REAL EXCEPT ITEMS ;
```

analyzes all non-dummy members, excluding those of type ITEM.

Note:

DUMMY is accepted in conjunction with ALIASES and/or CATALOGS but has no effect on the processing of these index-name types.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of making selections based on member/index-name type.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of secondary selections.

Analyzing Member/Index-names in a KEPT-DATA List, Locked Members, or Particular Members

To analyze member/index-names in an unnamed KEPT-DATA list, enter:

```
ANALYZE KEPT-DATA ;
```

To analyze member/index-names in a named KEPT-DATA list, enter:

```
ANALYZE KEPT-DATA IN list-name ;
```

where *list-name* identifies a particular named KEPT-DATA list.

To analyze only those encoded members which are locked, enter:

```
ANALYZE LOCKED ;
```

To analyze one or more named members, enter:

```
ANALYZE MEMBERS member-name-list ;
```

where *member-name-list* is one or more member names separated by commas.

Member/index-names in a KEPT-DATA list are analyzed in the order in which they appear in the KEPT-DATA list. Members in a list of member names are analyzed in the order they are specified in the member-name-list.

If you specify the keyword ALPHABETICALLY in the command, the member/index-names are output in alphanumerical order.

For example:

```
ANALYZE MEMBERS EMP-CODE, ACC-RPT, ACC-CODE ALPHABETICALLY ;
```

outputs logical block utilization details of ACC-CODE then ACC-RPT followed by EMP-CODE.

Locked members are automatically output in alphanumerical order.

If you specify KEPT-DATA and/or LOCKED and/or MEMBERS *member-name-list* in the command, then all member/index-names that fall within any of the selection criteria included in the command are analyzed. Member names that appear in more than one of the categories are processed only once.

For example:

```
ANALYZE KEPT-DATA IN ADDRESS MEMBERS EMP-CODE, ACC-RPT ;
```

analyzes the member/index-names in the KEPT-DATA list named ADDRESS followed by the members EMP-CODE and ACC-RPT.

You can refine the above variants of the ANALYZE command to exclude specific types of member/index-names from being analyzed by using EXCEPT.

For example, the command:

```
ANALYZE KEPT-DATA EXCEPT FILES, GROUPS ;
```

analyzes all member/index-names held in the KEPT-DATA list, excluding those of the types FILE and GROUP.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of the KEEP and ALSO KEEP commands.

Analyzing Particular Types of Member/Index-name

To analyze all member/index-names, enter:

```
ANALYZE INDEX-NAMES ;
```

To analyze specified types of member/index-names, enter:

```
ANALYZE member/index-name-type ;
```

where *member/index-name-type* is the interrogate keyword for any member type, or any collective member type, available in your dictionary, or the keywords ALIASES, CATALOGS, and (if the User Defined Syntax facility, selectable unit CMR-UD1, is installed) ATTRIBUTES. When more than one member/index-name-type is specified each must be separated by a comma.

To analyze all member/index-names, except those of a specified type, enter:

```
ANALYZE INDEX-NAMES EXCEPT member/index-name-type ;
```

To analyze the aliases of all encoded members, enter:

```
ANALYZE ALIASES ;
```

To analyze the catalog classifications of all encoded members, enter:

```
ANALYZE CATALOGS ;
```

To analyze the user defined indexed attributes of all encoded members, enter:

```
ANALYZE ATTRIBUTES ;
```

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for a definition of member/index-names.

Excluding Particular Types of Member/Index-name from the Command

Refer to "Analyzing Particular Types of Member/Index-name" on page 92 of this command specification for details of how to exclude particular types of member/index-name from being analyzed.

Selecting What You Want to Process by Status, Name, Time, or User

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of secondary selections.

Controlling Output from an ANALYZE Command

To analyze the whole dictionary and output a summary of the logical block utilization, enter:

```
ANALYZE GIVING CONTROLS ;
```

To analyze the whole dictionary and output a list of each type of member/index-name present in the dictionary, the total number of members of each type, and a summary of the logical block utilization, enter:

```
ANALYZE GIVING TOTALS ;
```

To analyze the whole dictionary and output the logical block size details of every member in a dictionary, a list of each type of member/index-name present in the dictionary, the total number of members of each type, and a summary of the logical block utilization, enter:

```
ANALYZE GIVING DETAILS ;
```

To restrict the analysis to those member/index-names which utilize more than a certain number of logical blocks, enter:

```
ANALYZE GIVING DETAILS OVER n ;
```

where *n* is an unsigned integer made up of the total number of logical blocks used for the member's source and data entries records across all statuses.

Details of what the output from these commands contains are provided:

- ANALYZE GIVING CONTROLS
- ANALYZE GIVING TOTALS
- ANALYZE GIVING DETAILS

Output of the ANALYZE GIVING CONTROLS Command

The ANALYZE GIVING CONTROLS command outputs a summary of the logical block utilization details for the whole dictionary.

The output appears under the sub-heading DICTIONARY BLOCK USAGE and includes this logical block utilization details for each dataset:

- The logical block size in bytes
- The number of logical blocks used
- The number of free logical blocks.

The number of logical blocks of the data entries dataset that are used for encoded records, and for various other internal purposes, are also given.

Example output is illustrated in Figure 7:

Figure 7 • Example Output

```
DICTIONARY BLOCK USAGE

DATASET         BLOCK SIZE          USED BLOCKS     FREE BLOCKS
SOURCE          800                 1116            534
INDEX           800                 65              40
DATA ENTRIES    800                 1221            679


                1 BLOCK USED BY DATAMANAGER
                1 BLOCK USED FOR ALIAS TABLES
                14 BLOCKS USED BY SECURITY
                1 BLOCK USED BY STATUS
                79 BLOCKS USED BY OFFSET TABLES
                1125 BLOCKS USED BY ENCODED RECORDS
```

Output of the ANALYZE GIVING TOTALS Command

The ANALYZE GIVING TOTALS command outputs the logical block utilization details for the whole dictionary.

The output appears under two sub-headings: ANALYSIS TOTALS and DICTIONARY BLOCK USAGE.

Output under the sub-heading ANALYSIS TOTALS includes these details for each type of member/index-name, for both the source and data entries datasets:

- The number of entries
- The number of logical blocks used
- The average number of logical blocks per entry
- The percentage of entries occupying one, two, three, and so on logical blocks, with the average percentage utilization of the last logical block occupied

The total number of member/index-names is also printed, against the member/index-name category (UNIQUE). (This total may differ from the number of entries in the source and data entries datasets, due to members with records in two or more statuses.)

Output of the ANALYZE GIVING DETAILS Command

The ANALYZE GIVING DETAILS command outputs the logical block utilization details for the whole dictionary.

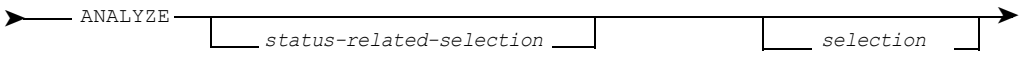
The output appears under three sub-headings: MEMBER ANALYSIS, ANALYSIS TOTALS, and DICTIONARY BLOCK USAGE.

Output under the sub-heading MEMBER ANALYSIS includes an analysis of the logical block utilization for each member/index-name individually. This analysis has a separate line for each status in which the member/index-name has a record, showing:

- Status information
- The member/index-name category
- The number of logical blocks (to two decimal places) used in each of the source and data entries datasets.

Complex Analyses

If you want to perform a complex analysis of logical block utilization including one or more selections and/or one or more secondary selections, use this format:



Several status-related-selection criteria and/or several time-and-user-related-selection criteria can be combined in the ANALYZE command (together with any selection criteria and a GIVING clause) to form complex analyses.

For example, to analyze all member/index-names in a KEPT-DATA list which have been AMENDED or REVERIFIED, enter:

```
ANALYZE AMENDED, REVERIFIED KEPT-DATA ;
```

If, subsequently, you want to refine the operation of the analysis to only those members whose names start with the character string PROG, and which were entered in the dictionary before the 31st May 1999, by the user (or department) whose password for the AUTHORITY command is PRODUCTION, enter:

```
ANALYZE AMENDED, REVERIFIED KEPT-DATA ONLY PROG IF INTRODUCED BY
                            PRODUCTION BEFORE '31 MAY 99' ;
```

You can further tailor the output of the command by using the GIVING clause.

For example, to output logical block usage details for each of the individual members analyzed, enter:

```
ANALYZE AMENDED, REVERIFIED KEPT-DATA ONLY PROG IF INTRODUCED BY
                 PRODUCTION BEFORE '31 MAY 99' GIVING DETAILS ;
```

Examples

```
ANALYZE GIVING TOTALS ;
```

analyzes the whole dictionary, and outputs logical block utilization details under the sub-headings: ANALYSIS TOTALS and DICTIONARY BLOCK USAGE.

```
ANALYZE IF AMENDED BY ACCOUNTS GIVING DETAILS ;
```

analyzes all member/index-names which have been updated by the user (or department) whose password was entered as ACCOUNTS in the AUTHORITY command, and outputs logical block utilization details under the sub-headings: MEMBER ANALYSIS, ANALYSIS TOTALS, and DICTIONARY BLOCK USAGE.

To analyze all index-names whose records, in any status, occupy a total of 11 or more logical blocks in the Source and Data Entries datasets, enter:

```
ANALYZE INDEX-NAMES GIVING DETAILS OVER 10 ;
```

To analyze all index-names whose names appear in the unnamed KEPT-DATA list, except those of the type FILE, enter:

```
ANALYZE KEPT-DATA EXCEPT FILES ;
```

To analyze the four listed members, enter:

```
ANALYZE MEMBERS EMP-CODE1, ACC-RPT, EMP-CODE2, ACC-SUB ;
```

To analyze all locked index-names created by the user PERSONNEL before the 31st of May 1998, enter:

```
ANALYZE LOCKED IF INTRODUCED BY PERSONNEL BEFORE '31 MAY 98' ;
```

To analyze all members with a source record, enter:

```
ANALYZE SOURCES ;
```

To analyze all index-names which include the character string EMPLOYEE in their names, enter:

```
ANALYZE WHEN ANY EQ EMPLOYEE ;
```

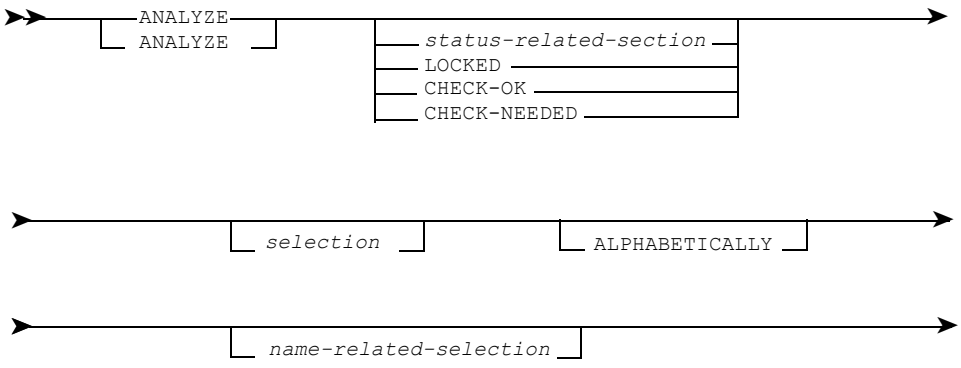
To analyze all index-names in the unnamed KEPT-DATA list that have a source record in the current status and in at least one of its base statuses, enter:

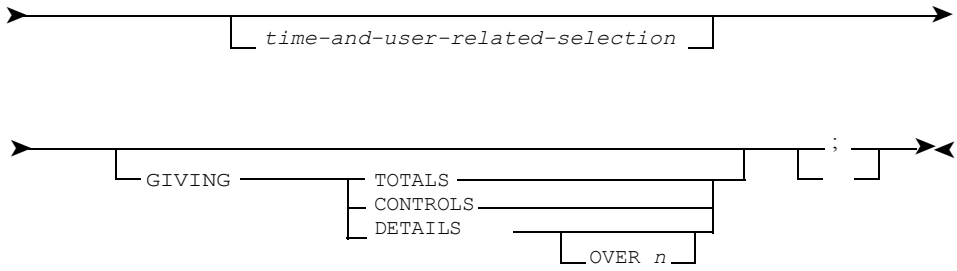
```
ANALYZE AMENDED KEPT-DATA ;
```

To restrict this analysis to those members whose names start with the character string PROG, enter:

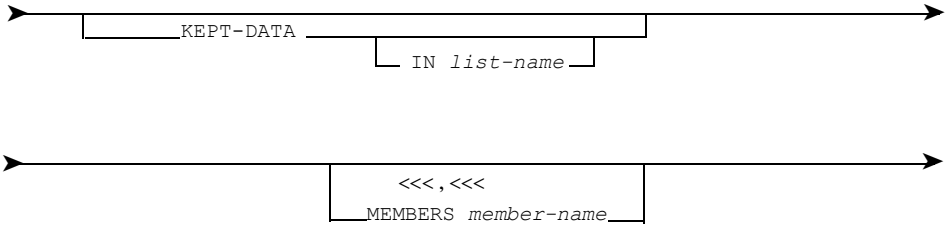
```
ANALYZE AMENDED KEPT-DATA ONLY PROG ;
```

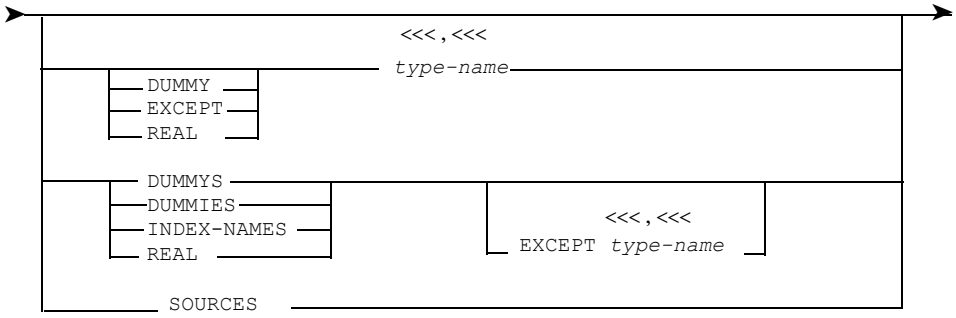
ANALYZE Syntax





where *selection* is:



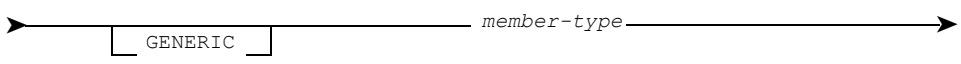


where:

list-name is the name of a KEPT-DATA list.

member-name is the name of a repository member.

type-name is:



AUDIT

The AUDIT command enables you to produce reports of repository usage that can be comprehensive or very specific. Usually you audit the repository after you have opened it using the Controller's authority. However, you can also run an audit from outside the repository to produce a batch file that you can use to re-input selected commands.

Refer to "AUDIT Syntax" on page 108 for the syntax of the AUDIT command.

You can produce audit reports for the current or an archived log dataset, based on:

- All commands
- A single command, selected by the transaction number allocated to the command when it was logged,

and reports based on:

- Updates
- Enquiries
- Return codes
- Named commands

can be further limited using selections based on:

- Status
- User
- Date and/or time
- A range of transactions

The keyword OF can optionally be included, but if you do specify OF, then you must either name a command or specify a single transaction.

Audit selections must be entered in the order shown in the syntax. However, selections by a group of transactions, a time, or from the archived dataset can be in any order provided they are the last elements in the AUDIT command.

The AUDIT command itself is not logged.

Refer to Chapter 4, "Security Facilities," on page 9 for details of commands that are logged.

Refer to Chapter 5, "Logging," on page 13 for details of uses of auditing.

Auditing the Whole Log

To output an audit report of every transaction recorded in the repository's log dataset, enter:

```
AUDIT ALL-COMMANDS ;
```

If you simply enter:

```
AUDIT ;
```

then the ALL-COMMANDS keyword is assumed by default. Output from these commands can be lengthy.

Auditing Enquiries

To produce an audit report of commands that interrogate the repository, enter:

```
AUDIT ENQUIRIES ;
```

If you keep a log of updating commands only, the JOURNAL and UNLOAD commands are reported.

Auditing Updates

To produce an audit report of commands that update the repository, enter:

```
AUDIT UPDATES ;
```

You can report on logged updating commands according to their recorded result code.

To produce an audit report of updates that were successful and so have a result code of 0, enter:

```
AUDIT RCO-UPDATES ;
```

You can use the generated report to find out if any incorrect changes have been made to members. To maintain close control over the repository and ensure that all source records have adequate safeguards, produce audit reports of successful updating commands on a day-to-day basis.

To produce an audit report of updates that were unsuccessful and so have a result code of 4, enter:

```
AUDIT RC4-UPDATES ;
```

Auditing RC4 updates after running commands in batch mode overnight means you do not have to do extensive manual checking for failed updates. If messages are logged, you can use *ASG-Manager Products Messages Guide* to determine why particular commands failed.

The number of failed updates can indicate how efficiently the repository is used, and can be particularly useful as a training aid. For example, if you only log error messages and updating commands the report includes:

- The JOURNAL command
- The UNLOAD command
- Any unsuccessful non-updating commands

You can find out from the audit report the frequency and type of errors made by a particular user and, if the errors indicate lack of understanding in particular areas of usage, the user can be given further instruction.

To produce an audit report of updates that were successful, but inhibited the roll-forward capability of the log and so have a result code of 8, enter:

```
AUDIT RC8-UPDATES ;
```

It is useful to know the number and frequency with which updating commands inhibit roll-forward capability. You can reconsider repository usage and backup procedures accordingly. Using the information from the audit report you can aim to reduce or eliminate the number of result code 8 commands processed.

Auditing Specific Commands

To produce an audit report of all the instances that a specific command was used, enter:

```
AUDIT  'command' ;
```

where *command* is a command identifier keyword, or a valid truncation of it, within delimiters.

This enables you to produce audit reports to analyze the use of specific command. For example, you can find out about changes in the protection labels of members.

If a command is renamed (using command tailoring functions), you must specify the user-defined name of the command, or a non-ambiguous truncation of it, in the AUDIT command.

If a command is renamed part way through the period covered by the report you can specify either name. The report will include these things:

- Transactions of the command under its old name
- Transactions of the command under its new name

If a command is renamed differently for different users (using tailoring the environment facilities), the heading of the report gives the name of the command as known to you and includes all transactions in which it is used, as issued by the user.

For example, on the 1st of June the PRINT command:

- Is renamed as DISPLAY in your logon profile
- Is renamed as OUTPUT in USER-A's logon profile
- Is not renamed in USER-B's logon profile

To produce an audit report at the end of June that includes:

- All PRINT commands issued in May
- All PRINT commands issued by USER-B in June
- All DISPLAY commands issued by you in June
- All OUTPUT commands issued by USER-A in June

enter:

```
AUDIT 'DISPLAY' AFTER APR/99 ;
```

The report heading is:

```
AUDIT REPORT OF 'DISPLAY' COMMANDS AFTER APR 99
```

Auditing Specific Transactions

Commands are given a transaction number when they are logged. To find out what the transaction numbers on the log dataset are, use the LOG STATUS command.

Refer to Chapter 5, "Logging," on page 13 for details of the output of the LOG STATUS command. To produce an audit report on a single transaction, enter:

```
AUDIT TRANSACTION number
```

where *number* is the number of the transaction.

An audit of a single transaction cannot be refined using other selections. To produce an audit on a whole block of transactions, enter:

```
AUDIT FROM TRANSACTION n1 TO TRANSACTION n2 ;
```

where:

n1 is the number of a transaction on the log.

n2 is also the number of a transaction on the log and comes after *n1*.

Usually *n2* is greater than *n1*, but transaction numbers can wrap around. For example, transaction 999999 may be followed by transaction 1.

If you specify the first transaction only, it and all the following transactions up to the last on the log are included in the report.

If you specify the second transaction only, all transactions from the first on the log up to and including the one specified are included in the audit report.

The transaction numbers you specify must be on the dataset to be processed. This can be used with other selections, for example the command:

```
AUDIT RC4-UPDATES BY USER ACCOUNTS FROM TRANSACTION 1345 TO 1395 ;
```

produces an audit report of all failed updates issued by ACCOUNTS between transactions 1345 and 1395.

Auditing an Archived Dataset

To produce an audit report from a dataset that has been output onto magnetic tape using a LOG ARCHIVE command, enter:

```
AUDIT ARCHIVED ;
```

The generated report includes all transactions on the archived dataset, and enables you to trace what has happened to members, even after the log dataset has been archived.

You can refine the report; for example, the command:

```
AUDIT FROM 1 TO 40 ARCHIVED ;
```

reports transactions 1 through to 40 on the archived dataset.

Auditing a Closed Repository

To produce an audit report on a BDAM or VSAM repository that either cannot be or has not been opened, enter:

```
AUDIT DICTIONARY name pass ;
```

where *name* is the name of a repository and *pass* is the Controller's password. This variant of the AUDIT command cannot be used with a DIV repository.

Note:

You cannot use the BY USER or ONTO clauses with the DICTIONARY clause.

Auditing by Date, Time, User, and Status

You can use date, time, user, and status selections to further refine audit reports that are based on:

- UPDATES
- ALL-COMMANDS
- ENQUIRIES
- RCO-UPDATES
- RC4-UPDATES
- RC8-UPDATES
- The name of a command

To restrict an audit report to a specific status, enter:

```
AUDIT selection IN STATUS status-name ;
```

where *status-name* is the name of a repository status.

To restrict an audit report to a specific user, enter:

```
AUDIT selection BY USER user-name ;
```

where *user-name* is the name of a repository user.

To obtain audit reports of transactions performed by the Controller or Master Operator the user-name must be given as MASTER.

To restrict an audit report to a specific date, enter:

```
AUDIT selection ON date ;
```

where *date* is the date in any format accepted in your installation.

To restrict an audit report to transactions over a date period, enter:

```
AUDIT selection BETWEEN date AND date ;
```

To restrict an audit report to a specific time, enter:

```
AUDIT selection AT time ;
```

where *time* is the time in any format accepted in your installation.

To restrict an audit report to transactions over a time period, enter:

```
AUDIT selection BETWEEN time AND time ;
```

You can combine any or all of these selections if you want to produce a very selective report for a specific reason. For example:

```
AUDIT RC4-UPDATES IN STATUS PROD BY ACCOUNTS BETWEEN 03/JUL/99 AT
09:00:00 AND 25/APR/99 AT 13:00:00 ;
```

When a detailed investigation is necessary, for example if a security breach is suspected, the selective report produced enables you to focus onto a precise area of repository usage.

Refer to *ASG-ControlManager User's Guide* for details of date and time formats.

Directing Reports to a File

To direct the audit report output to a file that can be used as batch input into the repository, enter:

```
AUDIT command ONTO ddname ;
```

where *ddname* is the name of the file to which the report is output.

You can edit the file and selectively re-input those commands you need after a reload.

Layout of Output

This information is output for each transaction audited:

- The transaction number
- The date and time that the command started
- The user identification or, if the user was the Controller or Master Operator, the word MASTER
- The status name, if you have named statuses using status functions
- The command input lines, together with any associated source or amendment lines
- Any logged Manager Products messages produced by the transaction
- The date and time the command ended
- The physical input/output counts for each dataset. For a DIV dictionary, the counts reflect logical accesses to the shared buffer.
- The environment in which the command was issued, for example, a particular teleprocessing monitor, in batch mode, or by access call
- The identification of the central processing unit under which the transaction was carried out
- The command keyword BULK or PERFORM where the transaction was one of a series produced by either of these commands

For example, the command:

```
AUDIT OF TRANSACTION 11532 ;
```

produces a report such as that in Figure 1:

```
                        AUDIT REPORT OF TRANSACTION 11532
TRANSACTION:    11532
START DATE :    13 JUN 99 TIME: 16.21.24.00
USER NAME  :    ANALYST
STATUS NAME:    CONV
RESULT CODE:    4
                REMOVE 'TAX-CODE';
MESSAGE    :    90S VARIABLE 1: TAX-CODE
END DATE   :    13 JUN1999 TIME: 16.21.25
EXCP COUNT :    INDEX:2  SOURCE: 2  DATA-ENTRIES: 3  RECOVERY: 5
environment:    BATCH      CPU-ID  00020055
-----------------------------------------------------------------
DM124111        1 TRANSACTION(S) AUDITED
DM120091        AUDIT PROCESSING SUCCESSFUL
```

Figure 1. Audit Report of Transaction 11532

The report shows that the updating command REMOVE was logged as transaction 11532. The result code of 4 indicates that the command failed to update the repository. The message 90S indicates that the user, in this case ANALYST, had insufficient authority to remove the named member.

Examples

To produce an audit report of the repository usage of the user ANALYST, between transactions 11526 and 11531, enter:

```
AUDIT USER 'ANALYST' FROM 11526 TO 11531 ;
```

To produce an audit report of transactions carried out by user ANALYST in status CONV after 12:50 on 14th June 1999, enter:

```
AUDIT UPDATES IN STATUS 'CONV' USER 'ANALYST' AFTER 14/JUN/99 AT
12:50:00 ;
```

To produce an audit report of unsuccessful updating commands issued between 23:45 and 23:50 on the 15th of June, enter:

```
AUDIT RC4-UPDATES ON 15/JUN/1999 BETWEEN 23:45:00 AND 23:50:00 ;
```

To produce an audit report of successful updating commands issued between 17:00 on 12th June and 16:17 on 13th June, enter:

```
AUDIT RCO-UPDATES BETWEEN 12/JUN/99 AT 17:00:00 AND 13/JUN/99 AT
16:17:11 ;
```

To produce an audit report of commands that successfully updated the repository but inhibited the roll-forward capability of the log, enter:

```
AUDIT RC8-UPDATES ;
```

To produce an audit report of commands that interrogated the repository and were issued by the user ANALYST before 12:50 on the 14th of June, enter:

```
AUDIT ENQUIRIES BY USER 'ANALYST' BEFORE 14/JUN/99 AT 12:50:00 ;
```

To produce an audit report of all the times a PROTECT command was issued, since 12th June, enter:

```
AUDIT 'PROTECT' AFTER 12/JUL/1999 ;
```

To produce an audit report of all the times a REMOVE command was issued in status CONV on a log dataset that has been archived onto magnetic tape, enter:

```
AUDIT 'REMOVE' IN STATUS 'CONV' ARCHIVED ;
```

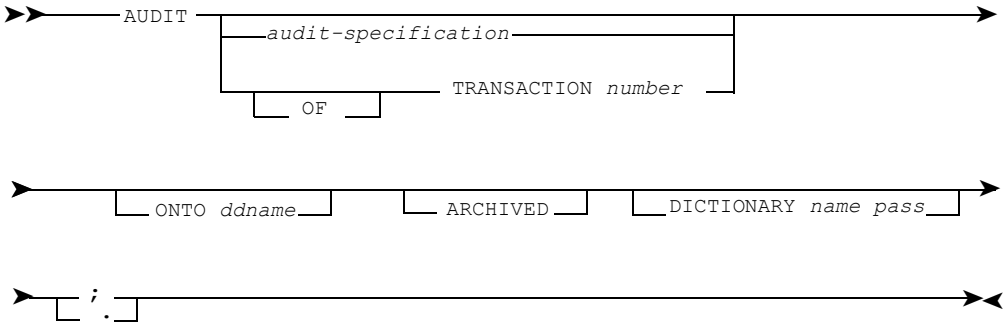
To produce an audit report of all the COPY commands issued by the repository Controller on an archived dataset, enter:

```
AUDIT 'COPY' USER 'MASTER' ARCHIVED ;
```

To produce an audit report of successful updates made by the user ACCOUNTS on 12th July 1999, and to direct the report to an external file, enter:

```
AUDIT RCO-UPDATES BY ACCOUNTS ON 12/JUL/99 ONTO RERUN ;
```

AUDIT Syntax

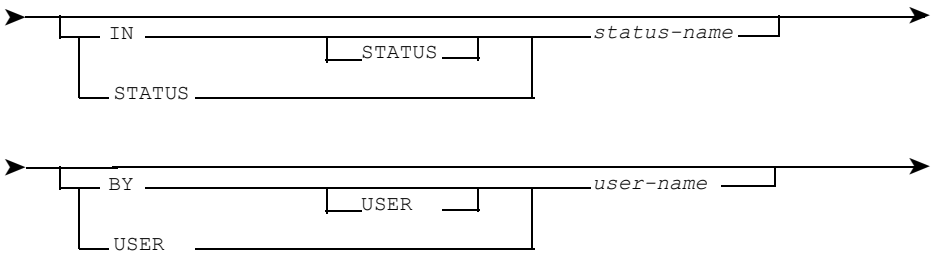


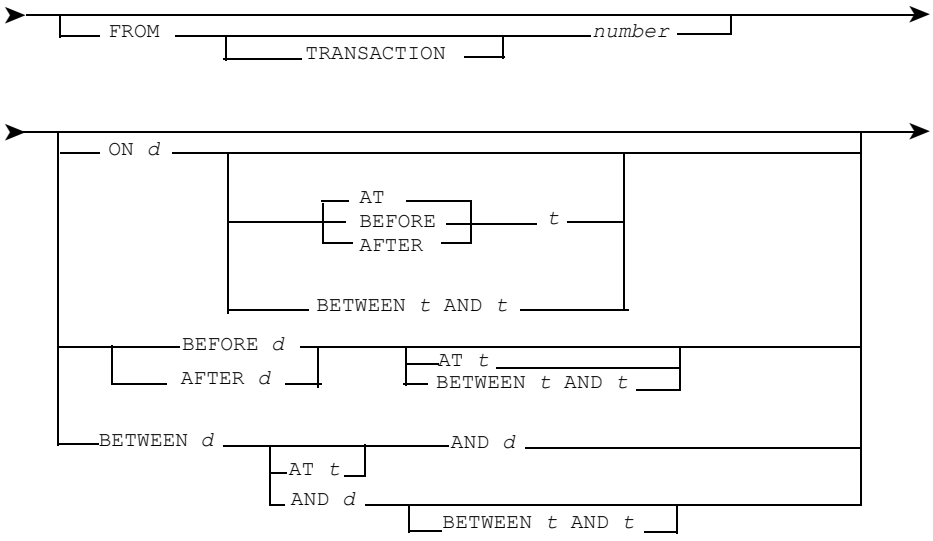
where *audit-specification* is:

where:

command-name is the command type identifier.

option is:





where:

status-name is the name of a status in the repository.

user-name is the name of a repository.

user number is the number of a transaction recorded in the repository's log.

d is a valid date.

t is a valid time (24 hour clock).

number is as defined above.

ddname is the ddname of the file to which output is directed.

name is the name of a repository.

pass is the Controller's password.

AUTHORITY

The AUTHORITY command is used to identify the user requiring access to the dictionary as the Controller; and/or to establish or change the master password, that is, the Controller's password; and/or to establish or change a password for the Master Operator.

Before an AUTHORITY command can be accepted, a dictionary must have been opened by means of a DICTIONARY command.

An AUTHORITY command must quote a password which has been registered in the dictionary by the Controller. Most Manager Products commands can only be entered after the AUTHORITY command has been successfully entered. These commands can be entered before the AUTHORITY command:

- DICTIONARY
- ENVIRONMENT
- CREATE
- VCREATE
- RELOAD
- ROLL-FORWARD
- LOG STATUS
- LOG ANALYSIS

All these commands, with the exception of DICTIONARY and ENVIRONMENT, are the Controller's private commands.

An AUTHORITY command quoting the master password must be entered before any other Controller's private commands can be accepted.

However, if an AUTHORITY command quoting the Master Operator's password is entered, these Controller's private commands can be accepted:

- UNLOAD
- LOG
- JOURNAL

The master password, that is the password for the Controller, and the Master Operator's password are registered in the dictionary by the Controller.

Note:

When you are working interactively from a hard copy terminal, you can prevent the password and the MASTER clause of an AUTHORITY command being printed by switching the terminal to full duplex operation.

If the MASTER clause and/or the OPERATOR clause is present in the command, the AUTHORITY command is treated as an updating command by the automatic recovery system.

Establishing a New Password for the Controller

To establish or change the master password (the Controller's password), enter:

```
AUTHORITY password MASTER new-password ;
```

where:

password is a character string of up to eight printable or non-printable characters, being the master password recorded in the dictionary.

new-password is a character string of up to eight printable or non-printable characters.

For example, to change a master password of non-printing characters to a more readily recognizable password, enter:

```
AUTHORITY
'               '
MASTER 'MSPDMR' ;
```

Note:

In a batch environment only the first line of the command is printed so that security is preserved.

If you use the MASTER variant of the AUTHORITY command *new-password* registers the password of the Controller. If a Master Operator's password is already registered, *new-password* replaces that password.

If the MASTER clause and/or the OPERATOR clause are present in an AUTHORITY command in which the password is a password recorded for a user other than the Controller, then the password is accepted as identifying that user and the MASTER clause and OPERATOR clause are ignored.

Establishing a New Password for the Master Operator

To establish or change the Master Operator's password, enter:

```
AUTHORITY password OPERATOR new-password ;
```

where:

password is a character string of up to eight printable or non-printable characters, being the master password recorded in the dictionary.

new-password is a character string of up to eight printable or non-printable characters.

If you use the OPERATOR variant of the AUTHORITY command *new-password* registers the password for the Master Operator. If a Master Operator's password is already registered, *new-password* replaces that password.

If the MASTER clause and/or the OPERATOR clause are present in an AUTHORITY command in which the password is a password recorded for a user other than the Controller, then the password is accepted as identifying that user and the MASTER clause and OPERATOR clause are ignored.

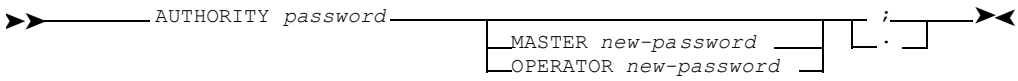
Format of Passwords in the AUTHORITY Command

Passwords are recorded in the dictionary as:

- 8-character fields
- Left justified with trailing spaces (hexadecimal 40)

For example, the passwords 'ABC' and 'ABC ' are treated as identical if the last three characters of 'ABC' are space characters, but not if they are other non-printing characters. The passwords 'ABC' and 'AB C', however, are not treated as identical.

AUTHORITY Syntax



where *password* is a character string of up to eight characters, being the master password recorded in the dictionary.

BULK ENCODE

The BULK ENCODE command is used to encode all or selected members of the repository. Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for documentation of the BULK ENCODE command. The OLD-DATE keyword is restricted to the repository Controller.

COMPARE UDS-TABLE

The COMPARE UDS-TABLE command is used to compare two UDS tables.

To compare two UDS tables in order to assess their compatibility, enter:

```
COMPARE UDS-TABLE proposed WITH current ;
```

where *proposed* is the name of a proposed UDS table and *current* is the name of the current UDS table.

You must issue this command before you issue a CONTROL UDS command to change the current UDS table.

A report is output showing how compatible the two named UDS tables are. The results of the comparison are stored on the MP-AID as a User Defined Syntax Comparison (UDSC) table. Where the results of the comparison indicate that the two UDS tables are incompatible, diagnostic messages are output which detail any incompatibilities. You may then choose to amend the new table, or implement it irrespective of any incompatibilities.

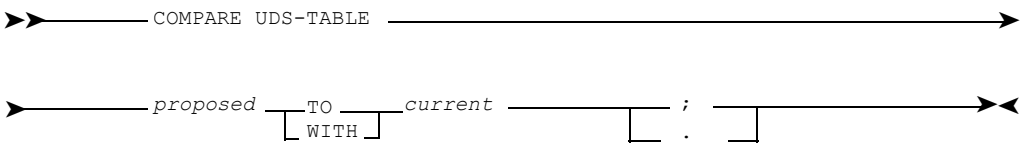
The command only compares UDS tables. It is your responsibility to ensure that the members in the repository are compatible with the new UDS table. For example, if you change the integrity rules, you may have to re-encode and alter some models to make them pass the altered integrity checks.

The command assumes the second table is the current table. Do *not* enter:

```
COMPARE UDS-TABLE current WITH proposed ;
```

since it may lead you unwittingly to make an incompatible table the current table.

COMPARE UDS-TABLE Syntax



where:

current is the name of the current UDS table proposed is the name of a proposed UDS table.

proposed is the name of a proposed UDS table.

CONSTRUCT UDS-TABLE

The CONSTRUCT UDS-TABLE command is use to validate a UDS HIERARCHY member held in a dictionary and to construct from it a UDS TABLE member on the MP-AID.

To validate a UDS hierarchy specified in a dictionary, and to construct from it a UDS-TABLE member on the MP-AID, enter:

```
CONSTRUCT UDS-TABLE FROM hierarchy ;
```

where *hierarchy* is the name of a HIERARCHY member in a dictionary (normally in *ASG-Manager Products Dictionary/Repository User's Guide*).

The validation procedures of the CONSTRUCT UDS-TABLE command ensure that the UDS-TABLE reflects a consistent and viable member-type structure. Where the hierarchy is found to be invalid, the UDS-TABLE is not CONSTRUCTed and an error message is output.

During processing of the CONSTRUCT UDS-TABLE command, the HIERARCHY member is automatically re-encoded if a SYNONYM clause or a UDO clause is generated as a result of the command.

The UDS Table Name

Where the HIERARCHY member contains an MP-AID-NAME clause, the name stated in that clause is given to the MP-AID UDS-TABLE member which results from a successful CONSTRUCT UDS-TABLE command.

Where the HIERARCHY member does not contain an MP-AID-NAME clause, then the first five characters of the HIERARCHY member name are used as the name of the UDS-TABLE on the MP-AID.

In either case, if the name of the new UDS-TABLE member would duplicate that of an existing UDS-TABLE member on the MP-AID, the CONSTRUCT UDS-TABLE command is rejected.

Where a UDS-TABLE name duplicates the name of a Manager Products load module, this will prevent the subsequent use of that load module.

The CONSTRUCT UDS-TABLE command removes any existing UDS Comparison tables which contain the name of the UDS table that is being CONSTRUCTed, as such tables will no longer be valid.

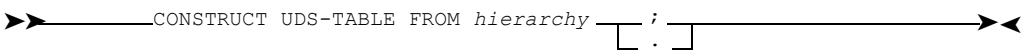
If you need to reCONSTRUCT a UDS-TABLE that is already in use in a dictionary, you will have to CONSTRUCT it under a different name; otherwise it would be impossible to COMPARE it and CONTROL it back into that dictionary. When a UDS table is changed in this way, a CONTROL UDS command quoting its new name must be issued for each dictionary that utilizes it.

Example of a CONSTRUCT UDS-TABLE Command

```
CONSTRUCT UDS-TABLE FROM PROJECT-INTEGRATION ;
```

In this example, if the HIERARCHY member PROJECT-INTEGRATION has an MP-AID name specified, that name will be the name of the resulting UDS-TABLE. If the HIERARCHY member does not have an MP-AID-NAME specified, then the name of the UDS-TABLE will be PROJE.

CONSTRUCT UDS-TABLE Syntax



where *hierarchy* is the name of a HIERARCHY member in a dictionary.

CONTROL CMS

The CONTROL CMS command enables you to initiate or to override concurrent usage protection for your dictionary in a CMS environment. The concurrent usage capability for your Manager Products dictionary must have been implemented as described in *ASG-Manager Products Installation in CMS Environments*.

To initiate concurrent usage protection, ensure that no other user is accessing the dictionary, then enter:

```
CONTROL CMS-USER user-id cuu cuu  ;
```

where:

user-id is the CMS user ID which has been allocated mini-disks to hold the dictionary files and dictionary control disks as defined in the implementation procedure.

cuu is a one-digit channel number and a two-digit unit number, being the address of a control disk. The first occurrence is the address of the serialization control disk; the second occurrence is the address of the dictionary utilization control disk.

To remove concurrent usage protection, ensure that no other user is accessing the dictionary, then enter:

```
CONTROL CMS-EXCLUSIVE user-id cuu cuu ;
```

where *user-id* and *cuu* are as defined above.

Examples

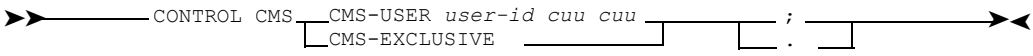
```
CONTROL CMS-USER AIDI001 194 195 ;
CONTROL CMS-EXCLUSIVE AIDI001 194 195 ;
```

Effect of Concurrent Usage Protection at Execution Time—Standard Usage

When the Manager Products enqueuing capability is enabled, all commands are ENQueued. This means that, when the enqueue capability is utilized, these conditions apply:

- If an update command is using the dictionary, then no command, whether an interrogation or an update, can use the dictionary concurrently. Thus, any commands submitted subsequent to the update command are put into a wait state and are queued according to order of submission.
- If an interrogation command is using the dictionary, then:
  - — If the next command submitted is an interrogation command, then the interrogation command is allowed concurrent usage of the dictionary. Any number of interrogation commands can be processed simultaneously.
  - — If the next command submitted is an update command, then the update command is put into a wait state. Any commands, whether interrogation or update, that are submitted after the update command are put into a wait state, and are queued behind the update command in order of submission.

CONTROL CMS Syntax



where:

user-id is the CMS user ID which has been allocated mini-disks to hold the dictionary files and dictionary control disks as defined in the implementation procedure.

cuu is a one-digit channel number and a two-digit unit number, being the address of a control disk. The first occurrence is the address of the serialization control disk; the second occurrence is the address of the dictionary utilization control disk.

CONTROL ENQ-NAME

The CONTROL ENQ-NAME specifies the minor queue name (that is, the *rname*) for the currently open dictionary, enter:

```
CONTROL ENQ-NAME 'string' ;
```

where *string* is a string of not more than 44 printable or non-printable characters.

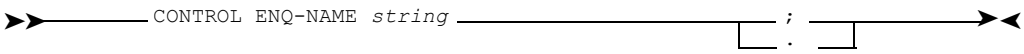
This command is only effective in OS environments for a BDAM or VSAM-organized dictionary. The command is rejected if issued for a DIV dictionary. It is accepted when DISP=OLD is specified in the dictionary job control statement. See the Manager Products installation publication appropriate to your environment for details of major and minor queue names.

The default name is generated as the dataset name of the index dataset when the dictionary is created. If the index dataset name is changed at any time, the change is not automatically reflected in the dictionary. This command should therefore be used whenever a dictionary's index dataset name is changed in OS environments, to bring the name recorded in the dictionary into line. The specified *rname* becomes effective on subsequently issued DICTIONARY commands.

Example

```
CONTROL ENQ-NAME 'DDICT.INDEX' ;
```

CONTROL ENQ-NAME Syntax



where *string* is a string of not more than 44 characters.

CONTROL NEW-ALIASES

The CONTROL NEW-ALIASES command is used to alter the alias table, to control the encoding of the ALIAS clauses of data definition statements, and to conform to your installation's particular requirements.

Note:

This version of the CONTROL NEW-ALIASES command supersedes use of the DALIAS macro supplied in Manager Products releases prior to ControlManager Version 2 Release 2.1.

The CONTROL NEW-ALIASES command enables you to define the specific alias-types which may be used at your installation in member type definitions. Using this command, you can also specify these things:

- Alias-type synonyms (that is, keywords which may be used in place of the alias-type in member definitions)
- Whether the use of a specific alias-type is optional or mandatory in member definitions
- The number of general aliases, and the total number of aliases, allowed for each member definition
- Rules which will be used for validating alias names when a new member is added to the repository. The rules may specify alias name length and/or whether alias names may be duplicates of other alias names or of member names.

Validation of a member definition against these rules takes place whenever a member is encoded. Checks for name duplication are also done whenever a new member is added to the repository (that is, during ADD, COPY, RENAME, or the generation of dummy members).

The total number of specific alias-types and general aliases must not exceed 254. The total number of alias keywords, including any synonyms, must not exceed 255. For new repositories, a default set of specific alias-types is provided. To display the contents of the alias table, use the command SHOW ALIAS-TYPES.

Adding an Alias-type

To add a new specific alias-type, enter:

```
CONTROL NEW-ALIAS ADD name-specification rules ;
```

where:

name-specification is the alias-type name, optionally followed by the keyword ELSE and an alias-type synonym. You can specify more than one synonym.

rules is the rules to be applied to this alias-type. (See below for information on applying rules to this alias-type definition.)

For example:

```
CONTROL NEW-ALIAS ADD LOCATION ELSE REGION ELSE AREA ;
```

adds alias-type LOCATION with synonyms of REGION and AREA. You can add more than one alias-type in a single ADD clause. For example:

```
CONTROL NEW-ALIAS ADD LOCATION ELSE REGION, NATURAL ;
```

adds alias-type LOCATION with a synonym of REGION, and also alias-type NATURAL.

If you attempt to add an alias-type and/or alias-type synonym which already exists, the command will fail.

Alias-type names should not be truncated to a point where they become ambiguous; nor should any complete alias-type name be the same as any truncated form of another alias-type name. The permissible extents of truncation of alias-type names should be publicized to users by the repository Controller.

Removing an Alias-type

To remove an existing alias-type and any associated synonyms, enter:

```
CONTROL NEW-ALIAS REMOVE name ;
```

where *name* is the alias-type name.

For example:

```
CONTROL NEW-ALIAS REMOVE LOCATION ;
```

You can remove more than one alias-type in a single REMOVE clause, for example:

```
CONTROL NEW-ALIAS REMOVE LOCATION, NATURAL ;
```

If you attempt to remove an alias-type which is used in an existing member definition on the repository, the command will fail.

If you attempt to remove an alias-type which does not exist, the command will fail.

Replacing/Modifying an Alias-type

To replace/modify an alias-type, enter:

```
CONTROL NEW-ALIAS REPLACE name-specification rules ;
```

where:

name-specification is an existing alias-type name, optionally followed by the keyword ELSE and an alias-type synonym. You can specify more than one synonym.

rules is the rules to be applied to this alias-type.

For example:

```
CONTROL NEW-ALIAS REPLACE ASSEMBLER ELSE ASM ELSE BAL ;
```

You can replace more than one alias-type in a single REPLACE clause, for example:

```
CONTROL NEW-ALIAS REPLACE ASSEMBLER ELSE BAL ELSE ASM, ADABAS ELSE
NATURAL ;
```

If the REPLACE removes existing alias-type synonyms, you will be warned when the alias table is updated. However, it is your responsibility to check the repository for use of the synonyms. Any synonyms must be either an existing synonym of the alias-type being replaced, or a valid alias-type keyword that is not already used as an alias-type or synonym.

If you attempt to replace an alias-type which does not exist, the command will fail.

Defining Name Validation Rules

To define name validation rules, enter them in a RULES clause in the CONTROL NEW-ALIAS command. Name validation rules may be specified for specific aliases, for general aliases, and for all aliases. If you specify name validation rules for all aliases, using the ALL-ALIASES keyword, these are taken to be default rules which will be applied to specific aliases for which you have not defined rules, and to general aliases if you have not defined rules explicitly.

The rules can include any or all of:

- Minimum length
- Maximum length
- Whether an alias name can be a duplicate of an existing member name, alias name of the same alias-type, or any other alias name used. You can choose whether each form of duplicate is allowed, not allowed, or whether the user will be warned. The default is that duplicates are allowed.
- Whether a user defined Executive Routine should be called to perform further name validation

For example:

```
CONTROL NEW-ALIAS ADD FOCUS RULES DUPLICATES MEMBERS ERROR ;
```

indicates that alias names for alias-type FOCUS may not be the same as names of existing repository members.

```
CONTROL NEW-ALIAS REPLACE FOCUS RULES MINIMUM-LENGTH 3 ERROR
MAXIMUM-LENGTH 30 ERROR ;
```

indicates that aliases for alias-type FOCUS must have a minimum of 3 characters and a maximum of 30.

If you specify a user-defined Executive Routine to perform validation, the alias name followed by the alias type is passed to the Executive Routine; for general aliases, the type is null. On completion, the encode fails if the return code is not 0. You may include error/warning messages for display to the user if the validation fails.

For example:

```
CONTROL NEW-ALIAS GENERAL-ALIASES MAXIMUM-NUMBER 10 RULES EXIT-EXEC
MYNAME ;
```

indicates that all general aliases will be validated by the user-defined Executive Routine MYNAME.

If you want to apply further validation when all aliases have been specified, you may use the encode exit to do so. See "SET ENCODE-EXIT" on page 202 for further information.

To suppress name validation, you may specify NO-RULES, for example:

```
CONTROL NEW-ALIAS ADD NATURAL NO-RULES ;
```

To apply the default rules specified in CONTROL NEW-ALIAS ALL-ALIASES, you may specify DEFAULT-RULES, for example:

```
CONTROL NEW-ALIAS ADD NATURAL DEFAULT-RULES ;
```

Optional and Mandatory Aliases

You can specify whether it is optional or mandatory that an alias is specified for a specific alias-type in the member definition, or whether the user will just be warned if the alias-type is not used. The default is that the alias is optional.

For example:

```
CONTROL NEW ALIAS ADD ASSEMBLER OPTIONAL NO ;
```

indicates that the user must include an alias for alias-type ASSEMBLER in every member definition.

```
CONTROL NEW-ALIAS ADD ASSEMBLER OPTIONAL WARNING ;
```

indicates that, if the user does not include an alias for alias-type ASSEMBLER in a member definition, a warning message will be displayed.

Limiting the Number of Aliases Used

You can specify the minimum number of general aliases to be used in each member definition. You can also specify the maximum number allowed; indeed, you must do so if you specify either a minimum number or name validation rules for general aliases, since you are replacing the entire definition for general aliases. The maximum number of general aliases may be raised provided this does not conflict with the number of specific alias-types.

If you do not use the GENERAL-ALIASES clause, the maximum number value is taken to be the previous maximum. For new repositories, use the SHOW ALIAS-TYPES command to see what is available by default.

You may use the ALL-ALIASES clause to specify the minimum and/or maximum number of all aliases used in each member definition, including general and specific aliases. If you do so, this number must be at least as large as any maximum number of general aliases you have specified.

For example:

```
CONTROL NEW-ALIAS GENERAL-ALIASES
    MINIMUM-NUMBER 1 WARNING MAXIMUM-NUMBER 10 ERROR
    RULES MINIMUM-LENGTH 3 ERROR MAXIMUM-LENGTH 32 ERROR ;
```

indicates that each member definition should include at least one general alias, and that the user will be warned if it does not (but the definition will be accepted). It also indicates that the maximum number of general aliases per definition is 10. Finally, the command includes name validation rules which require any general aliases used to have a minimum of 3 characters and a maximum of 32.

```
CONTROL NEW-ALIAS GENERAL-ALIASES MAXIMUM-NUMBER 20
    RULES DUPLICATES MEMBERS WARNING ALIASES ERROR ;
```

indicates that each member definition may include up to 20 general aliases. Further, the user will be warned if aliases are used which are duplicates of member names which exist on the repository, but they will be prevented from using an alias which is the same as any other alias name, general or specific.

Displaying the Alias Table

You can display the alias table at any time using the SHOW ALIAS-TYPES command.

To display the contents at the end of an update performed by the CONTROL NEW-ALIAS command, use the SHOW-ALIASES keyword:

```
CONTROL NEW-ALIAS ADD LOCATION SHOW-ALIASES ;
```

adds specific alias-type LOCATION, and then displays the new alias table.

If there are any errors when the command is executed, the alias table will not be displayed.

Testing Changes to the Alias Table

To test an update of the alias table without applying the changes, use the keyword QUIT. If you have specified SHOW-ALIASES, the alias table will be displayed, but the changes will then be abandoned, for example:

```
CONTROL NEW-ALIAS ADD FOCUS SHOW-ALIASES QUIT ;
```

adds specific alias-type FOCUS, displays the new alias table, and then abandons the update.

Impact of Validation On Other Commands

Validation carried out during RESTORE ALL will output warning messages instead of error messages.

Length checking and checks for mandatory alias-types in member definitions during a STATUS restructuring command will output warning messages instead of error messages. However, in both RESTORE ALL and STATUS, the warning messages will not show unless there is also an error message; printed Encode Output is suppressed for all but errors. It is important to check changes to the Alias Table at the point of update, when warnings will be output for any rules that are arrowed.

Errors can be found by using commands. Use a command like this to find members with the short alias missing:

```
WHICH MEMBERS NOT HAVE SHORT ALIAS SPEC ;
```

Use a command like this to find aliases that are too long:

```
GLOSSARY SHORT ALIAS WHEN LENGTH GT 4 ;
```

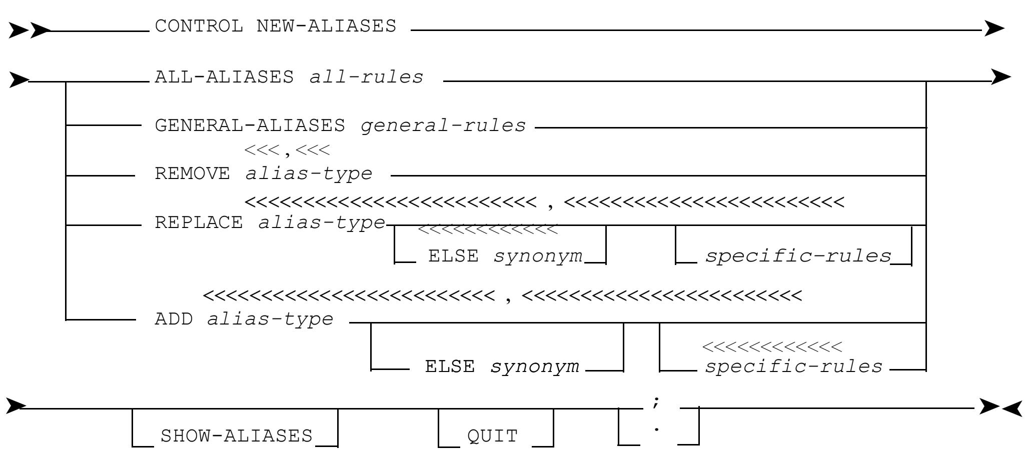
Checking for duplicates during the STATUS command will result in error messages. Checking during the STATUS command will take place across all visible statuses.

Syntax of Alias-type Keywords

Alias-type keywords are subject to these rules:

- They can be from a minimum of 1 character to a maximum of 32 characters in length
- They must start with a letter
- They must not contain any of these characters:
  - . (full stop or period)
  - ; (semicolon)
  - , (comma)
  - ((left parenthesis)
  -) (right parenthesis)
  - = (equals)
  - ' (single quote)
  - " (double quote)
  - space
  - lower case letters
  - non-printing characters

CONTROL NEW-ALIASES Syntax

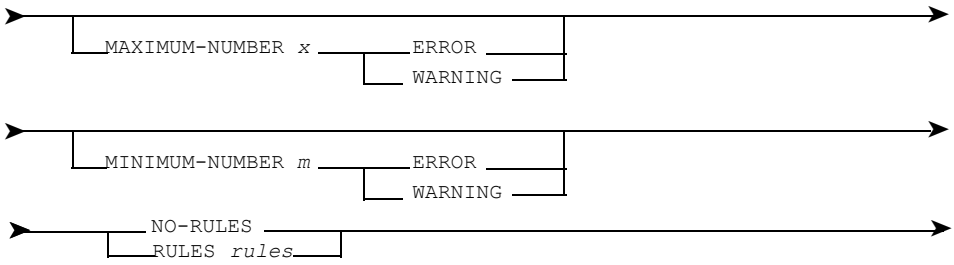


where:

alias-type is the name of a specific alias-type; naming rules are described above.

synonym is the name of an alias-type synonym (that is, an alternative name by which users may refer to an alias-type in member definitions).

all-rules is:

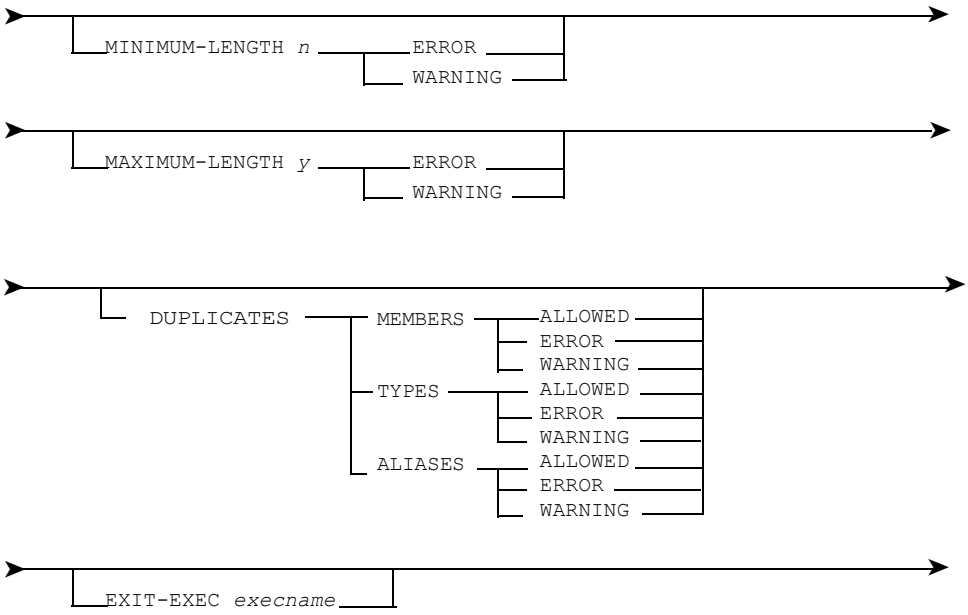


where:

x is an integer in the range 0 to 254

m is an integer in the range 1 to 254. *m* must be greater than or equal to *x*, if used.

rules is:



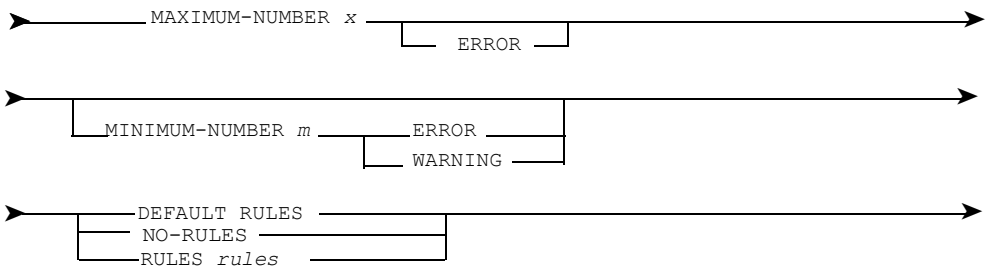
where:

n is an integer in the range 1 to 79.

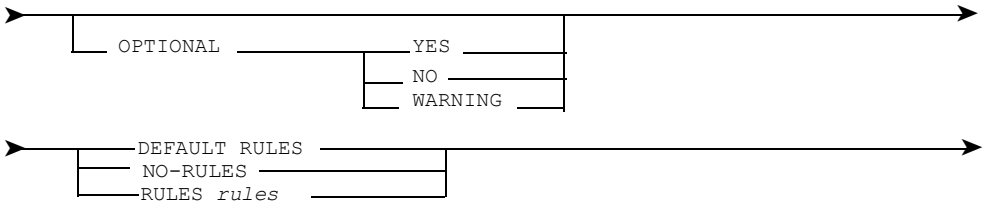
y is an integer in the range 1 to 79. *y* must be greater than or equal to *n*, if used.

execnname is the name of a user-defined Executive Routine and may be up to 10 characters in length.

general-rules is:



specific-rules is:



CONTROL RESERVE

A dictionary has a RESERVE control indicator in its control record. When this control indicator is set on for a currently open dictionary, only one central processor at a time can access that dictionary.

To set the RESERVE control indicator on, enter:

```
CONTROL RESERVE ON ;
```

This command is only effective in OS environments for a BDAM or VSAM-organized dictionary. The command is rejected if issued for a DIV dictionary. The command is only accepted when DISP=OLD is specified in the dictionary's job control statements.

To set the RESERVE control indicator off, enter:

```
CONTROL RESERVE OFF ;
```

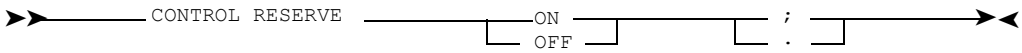
The RESERVE control indicator is set off when the dictionary is created.

See the appropriate Manager Products installation publication for details of concurrent dictionary usage.

Examples

```
CONTROL RESERVE ON ;
CONTROL RESERVE OFF ;
```

CONTROL RESERVE Syntax



CONTROL UDR

The CONTROL UDR command renames user-defined relationships, enter:

```
CONTROL UDR RELATIONSHIPS relationship1, relationship2
SUB-RELATIONSHIP sub-relationship
```

where *relationship 1*, *relationship2*, and so forth, are the names of the relationships you want to define in the dictionary, and sub-relationship is the sub-relationship which will apply to all the UDRs defined in this command. You may define up to nine UDRs in a CONTROL UDR command.

These names will rename the ASG-supplied default names (UDR1 to UDR9), in the order in which you specify them. Thus the first UDR you specify will rename UDR1, the second will rename UDR2 and so on. The same applies when you rename UDRs on any subsequent occasion; that is, each time you rename UDRs, you rename the default UDRs (UDR1 to UDR9).

The name you give to a UDR may not be the same as any of the relationship keywords which have a particular meaning in Manager Products software. When you enter a CONTROL UDR command, the UDRs are checked, and if any of the reserved keywords are among them, the command will fail.

The sub-relationship which you define will apply to all the UDRs specified in the CONTROL UDR command.

Manager Products Reserved Keywords

This is a list of the Manager Products reserved keywords which you must *not* use to define user-defined relationships.

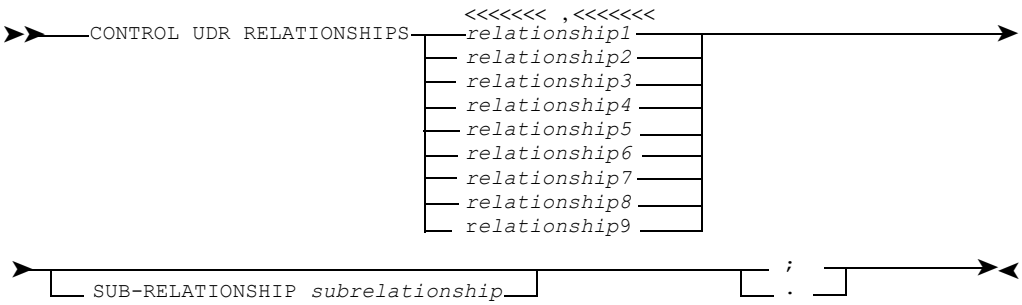
ACCESS	ACCESS-FILE	ACCESSES
ADD-TO	AFFECTS	ALL-SEGMENTS
ALLOW	AREA	AREAS
AS	ASSIGNED-TO	ATTRIBUTES
BATCH-GLOBAL	BOUND	
CALCULATED-USING	CALL	CALLS
CAUSES	CIPHER	CIPHER-BY
COMMBLOCK	COMMBLOCK-MEMBER	COMMON-ATTRIBUTES
COMMON-GLOBAL	COMPONENT-NUMBER	COMPOUND-KEY
COMPUTE	CONCATENATED-KEY-CONSTITUENTS	CONCATENATED-KEY-FIELDS
CONCATENATED-KEY-NAMES	CONNECT	CONNECT-TO
CONTAINS	CONTROL	COUNT-FIELD
COUNTS-AS	COUPLE	COUPLE-BY
CREATOR-OWNER		
DATABASE-KEYS	DATABASES	DATASETS
DELETES	DEPENDS-ON	DESCRIPTORS
DESTINATION	DISCONNECT	DISCONNECT-FROM
DL/1-DATASETS	DL1-DATASETS	DCML-AREA
DMCL-AS	DOMAIN	DUMMY
DUPLICATE-DATE-FIELDS		
EDIT-COMPRESSION-EXITS	EDIT-NAME	EDITPROC
ERASE	EVALUATE	EXIT-ROUTINES
FATHERS	FIELD-NAMES	FIELDPROC
FIND-USING	FIND	FOR-UPDATE
FOR	FOREIGN-KEYS	FROM
FUNCTION-USES		

GENERATES	GENERIC-ATTRIBUTES	GET
GIVING	GIVING-IN	GIVING-THROUGH
GRANTOR		
HAS		
IDENTIFIER	IF-SET	IF
IMPLEMENTS	IMPLIES	IMS-DATASETS
IN	IN-DATABASES	INCLUDES
INFLUENCES	INPUTS	
JOURNAL		
KEEP	KEY	KEYS
KEY-VALUES-FOR		
LAST-MENU	LAST-PANEL	LEFT-HAND-SIDE
LHS	LIKE	LINK
LINK-TO	LINKED-TO	LOCATED-IN
LOCATION	LOCATION-MODE	LOGICAL-CONTAINS
LOGICAL-CONTAINS-IN	LOGICAL-RECORD	
MAINTENANCE-EXITS	MASTER-PASSWORD	MEMBER
MODIFY	MULTI-ASSOCIATIONS	MULTI-ATTRIBUTES
NAME	NESTED	NEXT-PANEL
OBTAINS	OF	ON
ONE-ASSOCIATIONS	ONE-ATTRIBUTES	ONLINE-GLOBAL
ORDER-BY	OUTPUTS	OVERFLOW
OWNER	OWNER-AREA	
PARAMETERS	PARENT	PARENTS

PASSING	PERFORMS	PHONETIC-NAMES
PLACED-IN	PLACEMENT-SET	POPULATION-FOR
PRIME-MEASURE	PROCEDURE	PROCESSES-IDMS
PROCESSES-IMS-CONTAINS	PROCESSES-SESAM	PROCESSES-SQL
PROCESSES-SUDS	PROGRAM	
QUALIFIED-ON		
RANDOMISING-MODULES	RECORD	RECORDS
REFERENCES	RELIES-ON	RENAMES
RHS	RIGHT-HAND-SIDE	
SEARCH-KEY-FIELDS	SEARCH-KEYS	SECONDARY-KEYS
SECONDARY-SEQUENCE-ON	SEE	SEGMENT
SELECT-FROM	SELECT-MEMBERS	SELECT
SELECTING	SENSITIVE-FIELDS	SEQUENCE-KEY-CONSTITUENTS
SEQUENCE-KEYS	SET	SETS
SHARES-WITH	SHARING-WITH	SORT-KEYS
SORTED	SOURCE	SOURCE-SSR
SSAS	STAND-ALONE	STATE-IDENTIFIER
STATISTICS-FOR	STATISTICS-OF	STATISTICS-TO
STORAGE-GROUP	STORE	STORES-VIA
SUB-DESCRIPTORS	SUB-DOMAIN	SUB-ENTITIES
SUBSCHEMA	SUBSEQUENCE-FIELDS	SUFFIX
SUPER-DESCRIPTORS	SUPPORTS	SYMBOLIC-NAMES-FOR
TABLESPACE	TARGET	TARGET-SSR
TO		
UDO	UNIQUE-KEY-FIELDS	UPDATES
USED-IN	USER-EXIT	USER-LABELS
USER-PASSWORD	USER-PASSWORDS	USING

VALIDPROC	VIEW	VIEWS
VSAM-FILE		
WITH	WITHIN	

CONTROL UDR Syntax



where:

relationship 1-9 are the names of the relationships you want to define in the dictionary.

sub-relationship is the sub-relationship which will apply to all the UDRs defined in this command. You may define up to nine UDRs in a CONTROL UDR command.

CONTROL UDR REMOVE

The CONTROL UDR REMOVE command removes the user-defined relationships which currently apply in your dictionary.

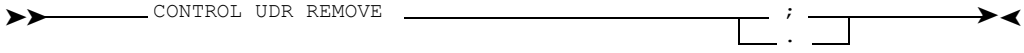
This is the format:

```
CONTROL UDR REMOVE ;
```

This command restores all user-defined relationships and sub-relationships to the default user-defined relationships and sub-relationships.

Before you remove UDRs from a dictionary, you will have to find out which members, if any, contain them, and delete them manually.

CONTROL UDR REMOVE Syntax



CONTROL UDS

The CONTROL UDS command implements a UDS table in a particular dictionary, enter:

```
CONTROL UDS table-name ;
```

where *table-name* is a one-to-five character name of a UDS table.

A CONTROL UDS command is accepted only from the Controller of an open dictionary.

ControlManager will initially assume *table-name* to refer to a UDS table held on the MP-AID; if the MP-AID does not hold such a UDS table then ControlManager will look for a corresponding UDS table held as a load module in the load library.

A CONTROL UDS command will only be accepted if:

- The dictionary is empty (that is, it has only just been CREATed), or
- The table name has previously been COMPAREd with the UDS table currently implemented in the dictionary.

The UDS table referred to by *table-name* replaces the UDS table currently implemented in the dictionary only if the result of a previously issued COMPARE command indicates that the two tables are compatible without re-encoding. Where no COMPARE command has been issued, or where the UDS tables are incompatible, a diagnostic message will be output.

You can force acceptance of a UDS table that has been flagged as incompatible by a COMPARE UDS-TABLE command, by entering:

```
CONTROL UDS table-name UNCONDITIONALLY ;
```

It would then be your responsibility to ensure that the contents of the dictionary conform to the new UDS table.

In online mode, if the command is successful, a message is output requesting you to issue the SHOW UDS-TABLE *table-name* FOR MSP variation of the SHOW command and send a copy of the result to ASG. In batch mode the SHOW UDS-TABLE *table-name* FOR MSP variant of the SHOW command is issued automatically by ControlManager.

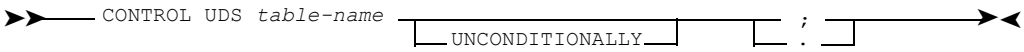
When a dictionary is first CREATED, the default UDS table is that provided in the ControlManager load module DU001.

The CONTROL UDS command automatically creates a secondary UDS-TABLE member on the MP-AID recording the usage of the UDS table.

Examples

```
CONTROL UDS PIH01 UNCONDITIONALLY ;
CONTROL UDS PIH01 ;
```

CONTROL UDS Syntax



where *table-name* is a string of between one and five characters, being the name of a UDS table.

COPY

The COPY command copies the source record of an existing member as the source record of a new member. It can also replace the source record of an existing member in the current status or in another status with the source record of another member.

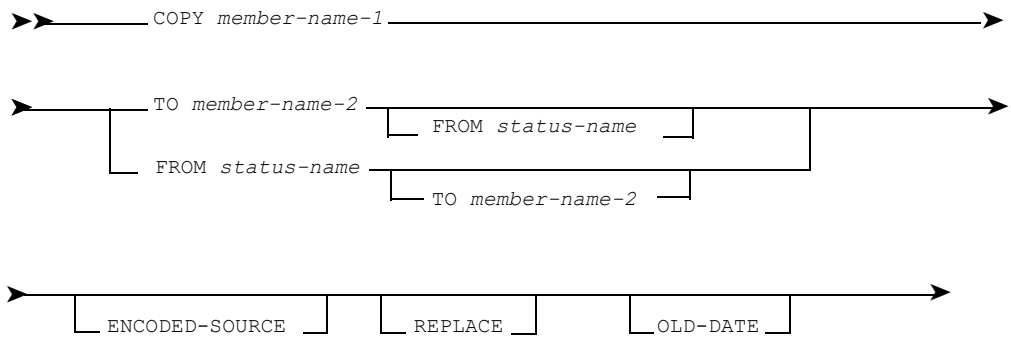
When a user issues a COPY command, the source record of the new member retains, as INSERTED and LAST ALTERED information, the date on which the copying activity took place, together with the logon ID of the user who issued the COPY command. If you, as Controller, issue a COPY command with an additional keyword, OLD-DATE, then the source record of the new member will instead retain the date and user information from the originating member. The OLD-DATE keyword thus allows you to maintain a historically accurate record of the evolution of your dictionary members.

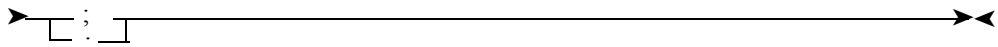
Examples

```
COPY ARTICLE-NUMBER TO COMPONENT-NUMBER OLD-DATE ;
COPY EMPLOYEE-NUMBER FROM CONV OLD-DATE ;
```

For details of the COPY command as available to all users, please refer to *ASG-Manager Products Dictionary/Repository User's Guide*.

COPY Syntax





where:

member-name-1 is the name of the existing dictionary member whose source record is to be copied.

member-name-2 is the name of the new dictionary member.

status-name is the name of a status.

CREATE

The CREATE command specifies the blocksizes of the datasets (files) that constitute the dictionary; initializes the dictionary with control information; assigns a master password to the dictionary; and determines whether a log is to be included as a fifth dataset of the dictionary.

To create a DIV dictionary for use under Manager Products Server Facility (MPSF), refer to the *ASG-Manager Products Server Facility User's Guide*.

If the Status facilities are installed, the command is used to specify the number of statuses required in the dictionary.

If the Audit and Security facility is installed, it is used additionally to specify the type of logging required.

The command is described in these items:

- CREATE: naming and initializing the dictionary and its datasets
- MASTER: assigning a master password
- STATUSES: specify the number of statuses in the dictionary
- LOG, UPDATES, ALL-COMMANDS: specifying if there is to be a log dataset and what will be logged
- PHYSICAL-BLOCKSIZE, IPB, SPB, DPB, RPB, LPB, VSAM: specifying the physical blocksize or blocksizes for a dictionary's datasets
- LOGICAL-BLOCKSIZE, DLB, ILB, SLB: specifying the logical blocksize or blocksizes for a dictionary's datasets.

Refer to "CREATE Syntax" on page 139 for the syntax of the CREATE command.

Naming and Initializing a Dictionary and its Datasets

To name and initialize a dictionary and its datasets, enter:

```
CREATE dictionary-name options ;
```

where:

dictionary-name is the name by which the dictionary will be known to users. The name can be up to six alphanumeric characters of which the first must be alphabetic. Space characters are not permitted.

options specifies the master password, logical and physical dataset blocksizes, level of logging, and number of statuses in the dictionary. A master password and physical and logical blocksizes must be specified.

The name declared for dictionary-name is taken as the name of the index dataset:

- The dictionary-name with the suffix S is taken as the name of the source dataset
- The dictionary-name with the suffix D is taken as the name of the data entries dataset
- The dictionary-name with the suffix E is taken as the name of the error recovery dataset
- The dictionary-name with the suffix J is taken as the name of the log dataset, if LOG is present in the CREATE command

The names of all four (or, if LOG is present in the CREATE command, all five) datasets must be declared in the job control statements for the operating system.

If there is any format error in the coding of the CREATE command, or if there is a discrepancy between the dictionary-name declared in the CREATE command and the name of the index dataset declared in the job control statements, an error message is output and the CREATE command is not actioned. If running under OS, the disk space for the four (or five) datasets remains allocated and named by the operating system.

Assigning a Master Password to a Dictionary

You *must* assign a master password to a dictionary.

To assign a master password include this clause in a CREATE command:

```
MASTER password
```

where *password* is a character string of up to eight printable or non-printable characters.

The master password defined in the CREATE command is that which identifies the Controller; other passwords are assigned to other individual users by SECURITY commands.

The master password must be quoted in an AUTHORITY command before any security information can be entered, changed, or listed by the OWNER or SECURITY commands and/or before any other of the Controller's private commands.

The master password can be changed, if this later becomes necessary, by a Controller's extension to an AUTHORITY command.

Specifying the Number of Statuses in a Dictionary

You can specify the number of statuses that are to be available in a dictionary.

To specify the number of statuses required in a dictionary, include this clause in a CREATE command:

```
WITH n STATUSES
```

where *n* is an unsigned integer in the range 1 to 255, specifying the number of statuses required in the dictionary.

The WITH clause is only accepted if the Basic Status or Advanced Status facility is installed (selectable unit CMR-DD2 or CMR-AD21). If the WITH clause is present in the command but a Status facility is not installed, an error message is output and the CREATE command is not actioned. If running under OS, the disk space for the four datasets remains allocated and named by the operating system.

If the WITH clause is present in the command and a Status facility is installed, the dictionary is created with the specified number of unnamed statuses. These statuses can be named and manipulated subsequently by STATUS commands.

If a Status facility is installed, and the WITH clause is omitted from the command, the dictionary is created with one unnamed status. This status can be named and manipulated subsequently by STATUS commands; but effectively, if there is only one status in a dictionary the Status facility is not used.

It is recommended that the value specified for *n* in the CREATE command be the maximum number of statuses required between the time the command is issued and the next time the dictionary is recreated. However, additional statuses incur costs in disk space and processing time, so no more should be specified than are really necessary.

Creating a Dictionary with a Log Dataset and Specifying What Will Be Logged

You can determine whether a log is to be included as a fifth dataset of the dictionary and specify the type of logging required.

To create a dictionary with a log include this clause in a CREATE command:

```
AND LOG
```

If LOG is present in the command, the dictionary is created with a fifth dataset, the log dataset. If LOG is omitted from the command, the dictionary is created with only four datasets.

If LOG is present in the command, all subsequent updating commands issued on the dictionary created by this command, together with their associated member definitions or amendments, are logged in the dictionary's log dataset.

The optional keyword AND introducing the LOG clause has no processing significance. It is available in the syntax purely for readability of the command.

The optional alternative keywords UPDATES and ALL-COMMANDS which are available with the LOG clause are accepted but ignored if the Audit and Security Facility (selectable unit CMR-DD3) is not included in your Manager Products configuration. If the Audit and Security Facility is installed, these keywords can be used to override the value of the COMTYPE parameter of the DLOG installation macro. (See the Manager Products installation publication relevant to your environment.)

If LOG is present but neither UPDATES nor ALL-COMMANDS is stated, then the value of the COMTYPE parameter of the DLOG installation macro determines whether the command operates as though LOG UPDATES were stated or as though LOG ALL-COMMANDS were stated.

If LOG UPDATES is stated, all subsequent updating commands issued on the dictionary created by this command, together with their associated member definitions or amendments, are logged in the dictionary's log dataset.

If LOG ALL-COMMANDS is stated, all subsequent commands issued on the dictionary created by this command, together with any associated member definitions or amendments, are logged in the dictionary's log dataset.

The option to log UPDATES or ALL-COMMANDS can be changed after the dictionary has been created, if required, using a variation of the LOG command.

Specifying Physical Blocksizes for a Dictionary

You must specify the size of the physical blocks in which the index, source, and data entries datasets will be held on disk. You can specify a blocksize for the recovery dataset but a default is provided if you do not. If logging is to be implemented you can specify a blocksize for the log dataset; a default blocksize is provided if you do not.

To specify individual blocksizes for each dataset (except if the dictionary is to be held in a VSAM dataspace), include these clauses in a CREATE command:

- IPB *nnnnn*
- SPB *nnnnn*
- DPB *nnnnn*
- RPB *nnnnn*
- LPG *nnnnn*

where *nnnnn* is the required dataset physical blocksize.

Valid abbreviations are used in the above clauses. These are the valid long forms of each abbreviation:

- IPB is a valid abbreviation of INDEX-PRYSICAL-BLOCKSIZE
- SPB is a valid abbreviation of SOURCE-PHYSICAL-BLOCKSIZE
- DPB is a valid abbreviation of DATA-PHYSICAL-BLOCKSJZE
- RPB is a valid abbreviation of RECOVERY-BLOCKSIZE
- LPB is a valid abbreviation of LOG-BLOCKSIZE

If you do not use the PHYSICAL-BLOCKSIZE keyword (see below) then IPB, SPB, and DPB must all be specified; no default blocksize is provided. RPB and LPB may be omitted if the default blocksize is acceptable.

To specify a common blocksize for all dictionary datasets (except if the dictionary is to be held in a VSAM dataspace), include this clause in a CREATE command:

```
PHYSICAL-BLOCKSIZE nnnnn
```

If the dictionary is to be held in a VSAM dataspace you must include this keyword in a CREATE command:

```
VSAM
```

The blocksize is specified using the RECORDSIZE parameter of the IDCAMS DEFINE CLUSTER commands. For VSAM dataspaces the equivalent of physical blocksize is specified using the RECORDSIZE parameter when defining the clusters via IDCAMS.

Specifying Logical Blocksizes for a Dictionary

You must specify the site of the logical blocks within the physical blocks of disk storage in which the index, source, and data entries datasets will be held. The logical blocksize of a dataset may not exceed its physical blocksize. The physical blocks of the log and recovery datasets are not divided into logical blocks.

To specify individual blocksizes for the datasets, include these clauses in a CREATE command:

```
ILB nnnnn
SLB nnnnn
DLB nnnnn
```

where *nnnnn* is the required dataset logical blocksize.

Valid abbreviations are used in the above clauses. These are the valid long forms of each abbreviation:

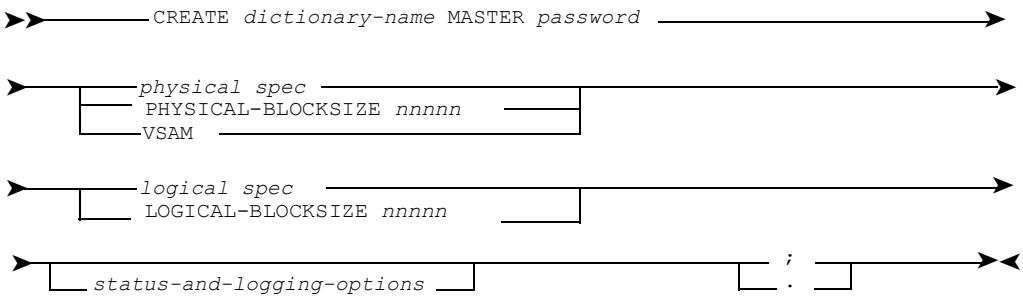
- ILB is a valid abbreviation of INDEX-LOGICAL-BLOCKSIZE
- SLB is a valid abbreviation of SOURCE-LOGICAL-BLOCKSIZE
- DLB is a valid abbreviation of DATA-LOGICAL-BLOCKSIZE

If you do not use the LOGICAL-BLOCKSIZE keyword, then ILB, SLB, and DLB must all be specified; no default blocksize is provided.

To specify a common blocksize for each of the relevant datasets, include this clause in a CREATE command:

```
LOGICAL-BLOCKSIZE nnnnn
```

CREATE Syntax

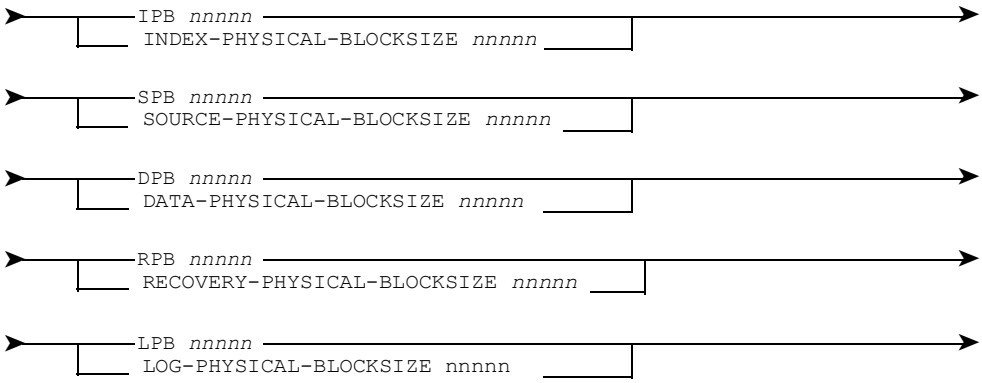


where:

dictionary-name is a string of up to six printable characters, being the name by which the new dictionary will be known.

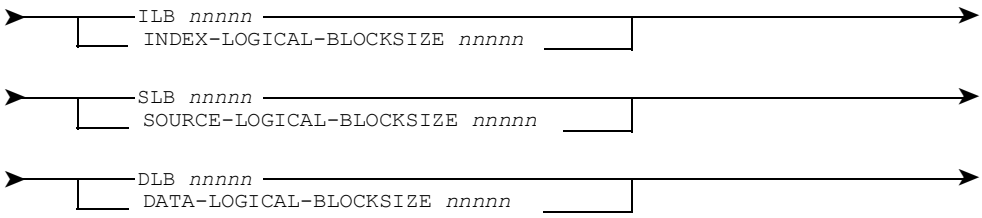
password is a string of up to eight printable characters.

physical-spec is:



where *nnnnn* is an integer specifying a logical or physical blocksize.

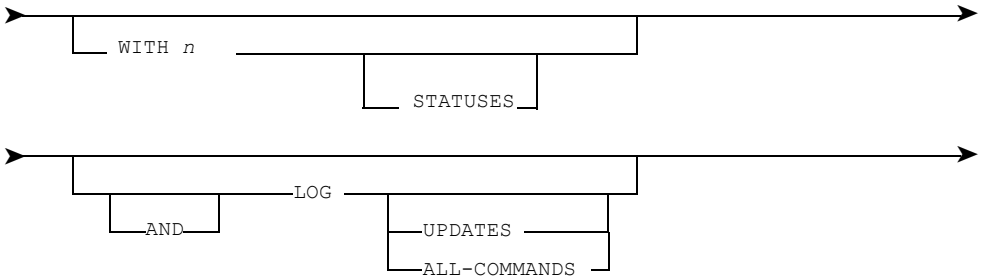
logical-spec is:



where:

nnnnn is as defined above.

status-and-logging-options is:



where *n* is an unsigned integer in the range 1 to 255.

DIAGNOSE

The DIAGNOSE command checks the consistency of the repository.

On any large disk-based system there is always the risk of data becoming corrupted. The diagnose facility enables you to check the consistency of your repository regularly, either as part of the backup process, or following a system crash to establish that the repository has not been corrupted.

As your repository gets larger, the time required to run a full diagnose will increase. You can check either the whole or selected parts of the repository to ensure that internal pointers or members' used and used-by cross references are consistent.

The diagnostic information output by the command enables ASG to advise you on how best to repair the corrupted repository. This is usually by using an XPRINT command but may be by using a RESTORE command.

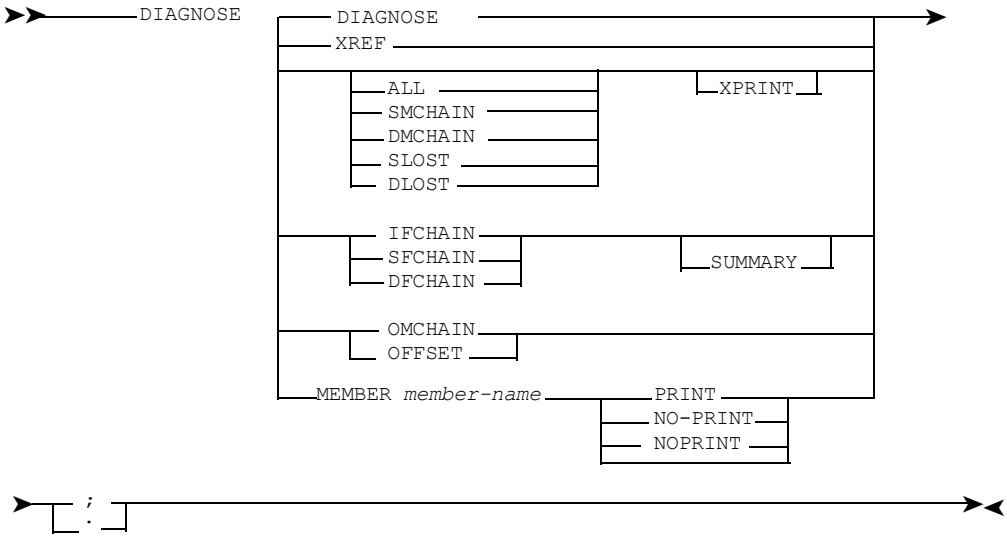
Note:

You cannot use the XREF keyword in CICS environments.

Keyword	Action
ALL	Checks the whole repository except the cross references
XREF	Checks that member references and usages cross refer
SMCHAIN	Checks each member's block chain on the source dataset
DMCHAIN	Checks each member's block chain on the data entries dataset
OMCHAIN	Checks for offset blocks that do not chain to any other blocks
SLOST	Checks for blocks on the source dataset that are not on any chain
DLOST	Checks for blocks on the data entries dataset that are not on any chain
IFCHAIN	Checks the index dataset free chain
SFCHAIN	Checks the source dataset free chain
DFCHAIN	Checks the data entries dataset free chain
MEMBER	Checks the references to and from an individual repository member

Keyword	Action
NOPRINT	Suppresses the printing of members that are correct
XPRINT	XPRINTs all the members and blocks that have errors
SUMMARY	XPRINTs the first 16 bytes of each block that has an error

DIAGNOSE Syntax



where *member-name* is the name of a repository member.

DICTIONARY

The DICTIONARY command opens a dictionary without opening the log dataset, enter:

```
DICTIONARY dictionary-name NOLOG ;
```

This capability could be required in order to perform an UNLOAD when the log dataset is corrupted or otherwise unavailable. The command opens the index, source, data entries, and error- recovery datasets. No further dictionary management commands are accepted until an AUTHORITY command quoting the master password, the Master Operator's password, or another DICTIONARY command is accepted.

The objective of DICTIONARY NOLOG is to enable an UNLOAD command to be issued. The issuing of other commands on a dictionary whose log dataset is inaccessible is not recommended.

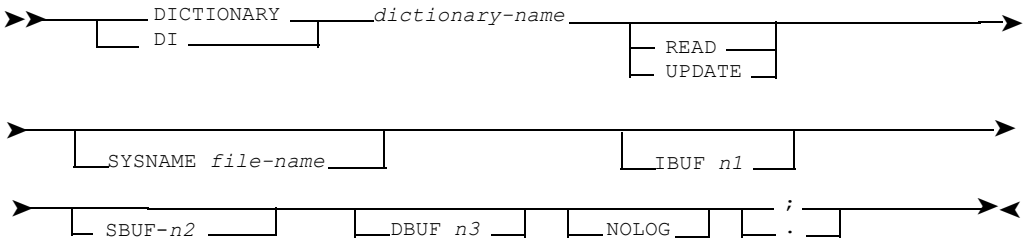
The dictionary is opened in update mode unless READ is specified. If READ is specified, UNLOAD will not be accepted.

Examples

```
DICTIONARY DEMO NOLOG ;
DICTIONARY INFO READ NOLOG ;
```

For details of the DICTIONARY command as available to all users, please refer to *ASG-Manager Products Dictionary/Repository User's Guide*.

DICTIONARY Syntax



where:

dictionary-name is the name of a dictionary that has been established by the Controller.

file-name is the logical file name (ddname) used in OS job control statements to indicate the external dataset name (physical file name) of the dictionary's index dataset.

n1, *n2*, and *n3* are unsigned integers in the range 1 to 32000.

For a DIV dictionary utilizing a single shared buffer, the keywords IBUF, SBUF, and DBUF are accepted but ignored. The size of the buffer is fixed and cannot be modified.

Refer to *ASG-Manager Products Server Facility User's Guide* for details of additional keywords required when opening the dictionaries for shared usage under MPSF.

DISABLE

To prevent the use of the dictionary by anyone other than the Controller or the Master Operator, enter:

```
DISABLE ;
```

When the DISABLE command is issued, a DISABLE flag is set in the currently open dictionary. Use of the dictionary by any person other than the Controller or the Master Operator is prevented while its DISABLE flag is set. The DISABLE flag is disabled by an ENABLE command issued when the dictionary is open under the Controller's or Master Operator's AUTHORITY.

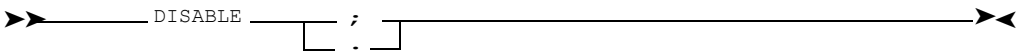
The setting of the DISABLE flag is not affected by closing and subsequent opening of the dictionary.

The DISABLE command is treated as an updating command by the automatic error recovery system.

Example

```
DISABLE ;
```

DISABLE Syntax



ENABLE

The ENABLE command cancels the effect of a DISABLE command. If a dictionary has been made unavailable to other users by a DISABLE command, you can make it available again to other users by entering:

```
ENABLE ;
```

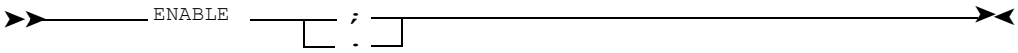
The ENABLE command disables the DISABLE flag in the currently open dictionary.

The ENABLE command is treated as an updating command by the automatic error recovery system.

Example

```
ENABLE ;
```

ENABLE Syntax



JOURNAL

The JOURNAL command allows you to write a transaction containing textual information to the log dataset.

The format for the command is:

```
JOURNAL 'string' ,'string' ;
```

where *string* is a character string of not more than 255 characters.

The JOURNAL command writes the whole command, exactly as input, as a transaction in the log dataset.

Although the JOURNAL command is not an updating command, it is not accepted if the dictionary is open in read-only mode, because update capability is needed in order to write to the log dataset.

The LOG BACKUP-DETAILS command writes to the control record of the log that a non-Manager Products backup has taken place, details of which are given in the Log Status report. The JOURNAL command should then be issued to record the backup as a transaction on the log dataset, details of which will be given in the Log Analysis report.

An UNLOAD command is written to the log as a transaction as well as automatically being written to the control record of the log. It is also automatically entered on the Log Status report.

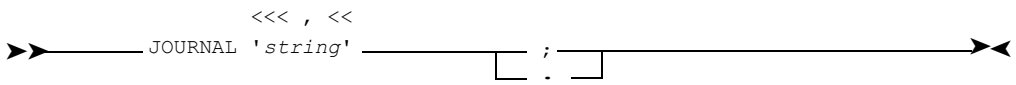
For example, to record the fact that the DCUST installation had been changed to increase the index buffers from 5 to 15, you would enter:

```
JOURNAL 'DCUST RETAILORED TO INCREASE 'INDEX BUFFERS'
                                      'FROM 5 TO 15' ;
```

To record when a backup was taken with non-Manager Products software, you would enter:

```
JOURNAL 'BACK-UP TAKEN OF DICTIONARY 1/2/99 AT 17.00';
```

JOURNAL Syntax



where *string* is a character string of not more than 255 characters.

LOCK RELEASE

The LOCK RELEASE command releases a lock on a member and makes the member available to all users for updating, enter:

```
LOCK RELEASE member-name ;
```

The LOCK RELEASE command is only accepted if entered by the user who last locked the member with a LOCK NEW, LOCK EXISTING, or LOCK RENEW command, or by the dictionary Controller, who can release any locks at any time.

If the LOCK RELEASE command is used to release either a current or an expired lock, that lock cannot be renewed with a LOCK RENEW command.

If a lock is not released with the LOCK RELEASE command, it will eventually expire when the lock period specified for the user, or for the dictionary, expires. The member will continue to be included in lists of locked members and the lock can be renewed (using the LOCK RENEW command) by the user who issued the lock which has expired.

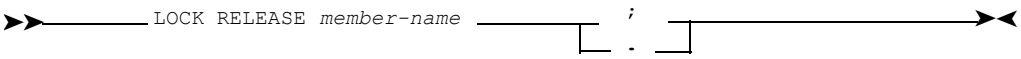
ASG recommends that the dictionary Controller check that expired locks, which are no longer required, are released (using the LOCK RELEASE command) on a regular basis. All members locked by a particular user can be released simultaneously with the PERFORM command. For example, the command:

```
PERFORM 'LOCK RELEASE "*"' LOCKED IF INTRODUCED BY
                           ACCOUNTS ON '31-MAY-98';
```

releases the locks on all locked members entered in the dictionary by the user (or department) whose password in the AUTHORITY command is ACCOUNTS on the 31st May, 1998.

If the Basic or Advanced Status facilities are installed (selectable units CMR-DD2 and CMR-AD21, respectively), the member whose lock is to be released must exist in the current status. If the member is currently locked in more than one status, only the lock in the current status is released.

LOCK RELEASE Syntax



where *member-name* is the name of a dictionary member.

LOG ALL-COMMANDS/UPDATE COMMANDS

This variation of the LOG command is used to initiate logging of all commands or dictionary updating commands. It is available only if the ControlManager Audit and Security facility (selectable unit CMR-DD3) is installed.

To initiate logging of all commands, enter:

```
LOG ALL-COMMANDS ;
```

To initiate logging of dictionary updating commands only, enter:

```
LOG UPDATE-COMMANDS ;
```

The dictionary Controller can revise the decision to log all commands or updating commands only, without having to recreate or reorganize the dictionary. The LOG ALL-COMMANDS or LOG UPDATE-COMMANDS command can be accepted whenever a dictionary with a log dataset is open. They apply only to the dictionary open when the command is issued.

These commands are accepted under the authority of the dictionary Controller or of the Master Operator. For the Controller, the command is immediately effective.

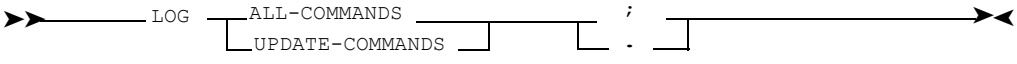
These commands are not logged, with the result that a subsequent ROLL-FORWARD command would not take account of them. The Log Status report contains an entry stating whether LOG ALL-COMMANDS or LOG UPDATE-COMMANDS is implemented.

For other users, the command does not become operative until the next DICTIONARY command is issued in respect of that dictionary. This applies for each user individually; that is, if the dictionary is in concurrent use by other users when the dictionary Controller issues the LOG ALL-COMMANDS or LOG UPDATE-COMMANDS command, the type of logging (all commands or only updating commands) in use continues for each user until that user issues a further DICTIONARY command for that dictionary. The type of logging specified in the Controller's command is then applied for that user.

LOG ALL-COMMANDS and LOG UPDATE-COMMANDS commands are treated as updating commands by the automatic error recovery system.

LOG ALL-COMMANDS records non-updating commands such as LIST, REPORT, QUERY, and so on in addition to updating commands.

LOG Syntax



LOG ANALYSIS

The LOG ANALYSIS command outputs an analysis of the information recorded in a dictionary's log dataset.

For the output from the LOG ANALYSIS command, please refer to "An Illustration of the Organization of the Log Dataset" on page 16.

LOG ANALYSIS—Usage

To analyze the current contents of the entire log, enter:

```
LOG ANALYSIS ;
```

To restrict the coverage of the analysis, the FROM and/or TO clause can be used. To do this, you would enter:

```
LOG ANALYSIS FROM TRANSACTION number TO TRANSACTION number ;
```

where *number* is the number of a transaction recorded in the dictionary's log. To output an analysis of the information recorded on a dictionary's archived log dataset, you would enter:

```
LOG ANALYSIS ARCHIVED ;
```

If the keyword ARCHIVED is present in the command, then instead of being produced from a dictionary's log dataset, the Log Analysis report is produced from a dataset previously output onto magnetic tape by a LOG ARCHIVE command.

To output the archived transactions 0 to 40, enter:

```
LOG ANALYSIS FROM 0 TO 40 ARCHIVED ;
```

To output the archived log when the dictionary is not open, enter:

```
LOG ANALYSIS ARCHIVED DICTIONARY DDICT 'MASTEROP' ;
```

Conditions for Use of the LOG ANALYSIS Command

The LOG ANALYSIS command is not logged.

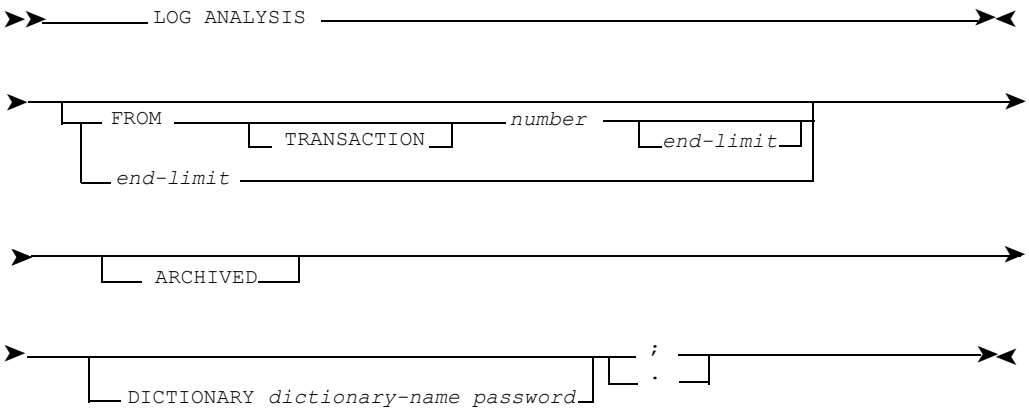
The LOG ANALYSIS command without a DICTIONARY clause is accepted only if a dictionary is open under the authority of the dictionary Controller or the Master Operator. If the command is accepted, the Log Analysis report for the dictionary is output.

The LOG ANALYSIS command with a DICTIONARY clause is accepted only if no dictionary is currently open. Unless ARCHIVED is also stated, the command opens the log dataset of the dictionary specified by dictionary name and checks whether the password quoted in the command is that of the dictionary Controller or the Master Operator. If it is not, the log dataset is closed and this error message displays:

```
DM12103E MASTER PASSWORD MISMATCH
```

If the quoted password is the password of the Controller or the Master Operator, the Log Analysis report for the dictionary is output. For a DIV dictionary, the DICTIONARY clause is not accepted unless ARCHIVED is also specified.

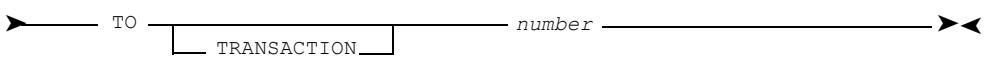
LOG ANALYSIS Syntax



where:

number is the number of a transaction recorded in the dictionary's log.

end-limit is:



where:

number is as defined above.

dictionary-name is the name of your dictionary.

password is either the Master Operator's or the Controller's password as recorded in the dictionary.

LOG ARCHIVE

The LOG ARCHIVE command archives to magnetic tape the contents of the primary area and/or the secondary area of the log dataset, enter:

```
LOG ARCHIVE ;
```

The LOG ARCHIVE command is not logged.

The LOG ARCHIVE command is accepted under the authority of the dictionary Controller or the Master Operator.

A log area can be archived only if its status is AWAITING ARCHIVE. If both the primary area and the secondary area are AWAITING ARCHIVE, then both areas are archived (in the correct historical sequence).

On acceptance of the first LOG ARCHIVE command following an UNLOAD command or a LOG BACKUP-DETAILS command, a new archive dataset is created containing (in addition to control information) only those transactions from the AWAITING ARCHIVE area of the log dataset. The status of the AWAITING ARCHIVE area of the log dataset is reset to AVAILABLE.

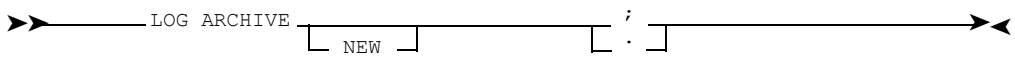
LOG ARCHIVE is treated as an updating command by the automatic error recovery system.

To force a new archive dataset to be started whether or not an UNLOAD has been performed since the last LOG ARCHIVE command, you would enter:

```
LOG ARCHIVE NEW ;
```

LOG ARCHIVE NEW is not normally recommended since it bypasses the reading of the previous archive dataset and thus does not provide early warning of tape input/output errors.

LOG ARCHIVE Syntax



LOG BACKUP-DETAILS

The LOG BACKUP-DETAILS command records in the log that a non-Manager Products backup of the dictionary has been taken, enter:

```
LOG BACKUP-DETAILS  'string' ;
```

where *string* is a character string of not more than 32 characters.

The LOG BACKUP-DETAILS command is accepted under the dictionary Controller's or the Master Operator's authority.

The LOG BACKUP-DETAILS command is not logged.

The specified string is recorded in the control record of the log dataset; it does not appear as a transaction. The JOURNAL command can be used to maintain a record of all backups taken, as transactions in the log.

The specified string appears in any subsequent Log Status reports for the dictionary, in the line:

```
LAST DICTIONARY BACK-UP TAKEN: string
```

until the next LOG BACKUP-DETAILS command or UNLOAD command is accepted.

A LOG BACKUP-DETAILS command should be issued immediately after each non-Manager Products backup is taken, in order to establish or maintain the roll-forward capability.

For example, if a backup is taken of transactions 1000 to 1999 and LOG BACKUP-DETAILS is then entered, the control record of the log dataset will contain information about the backup. If the dictionary then needs to be RELOADed from that backup it will roll-forward from transaction number 2000 by applying the current information on the log.

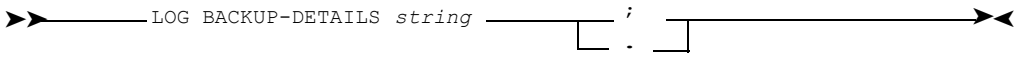
If, subsequent to this, transaction numbers 2000 to 3500 take place, and a backup is taken without LOG BACKUP-DETAILS being entered, the control record of the log dataset will not contain information about this last backup. Therefore, if the dictionary needs to be recovered and rolled forward, the log dataset control record will not contain any information indicating that transactions 2000 to 3500 have been backed up and do not need to be re-applied. There is also the possibility that the MAXEXCP parameter will be exceeded.

Although the LOG BACKUP-DETAILS command is not an updating command, it is not accepted if the dictionary is open in read-only mode, because update capability is needed in order to write to the log dataset.

For example, you would enter:

```
LOG BACKUP-DETAILS ''BY OPERATOR ON 881223 AT 17.00".
LOG BACKUP-DETAILS 'TAPE M50012 NOV15 06.30.45';
```

LOG BACKUP-DETAILS Syntax



where *string* is a string of up to 32 printable characters.

LOG CREATE

The LOG CREATE command recreates on a dictionary a log which is empty but which has the correct last transaction number, enter:

```
LOG CREATE DICTIONARY dictionary-name password ;
```

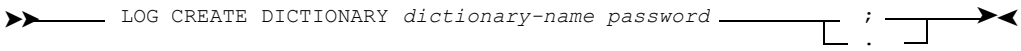
where *password* is the master password as recorded in the dictionary.

LOG CREATE is used when a log has been corrupted or destroyed but the rest of the dictionary is still intact. It will work whether or not the dictionary is already open and uses the index dataset of the dictionary to create a new log. When the command has been processed, the index dataset is closed, and a DICTIONARY command must be issued to open the dictionary.

The LOG CREATE command is not logged.

The LOG CREATE command is not available for use with a DIV dictionary.

LOG CREATE Syntax



where:

dictionary-name is the name of your dictionary

password is either the Master Operator's or the Controller's password as recorded in the dictionary.

LOG PURGE

The LOG PURGE command purges unwanted transactions from the log dataset, enter.

The format for this command is:

```
LOG PURGE DICTIONARY dictionary-name password ;
```

where *password* is the master password as recorded in the dictionary or archived log.

The LOG PURGE command is used when a ROLL-FORWARD has been accomplished up to a transaction (numbered or a result code 8) and it is decided to purge all the subsequent transactions.

The LOG PURGE command should be followed by a LOG ARCHIVE and an UNLOAD to prevent inhibition of subsequent roll-forwards.

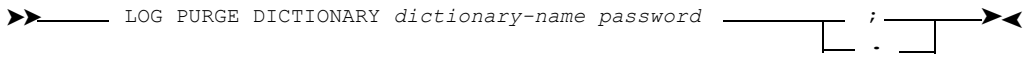
A LOG PURGE command is not logged.

The LOG PURGE command is not available for use with a DIV dictionary. Refer to the *ASG-Manager Products Server Facility User's Guide* for details of how to drop unwanted transactions from the log.

For example, to purge/remove all transactions after a ROLL-FORWARD to a result code 8, enter:

```
LOG PURGE DICTIONARY DDICT 'MSPDMR' ;
```

LOG PURGE Syntax



where:

dictionary-name is the name of your dictionary

password is either the Master Operator's or the Controller's password as recorded in the dictionary.

LOG STATUS

The LOG STATUS command ascertains the status and utilization of the log dataset, details of the last backup, and roll-forward information, enter:

```
LOG STATUS DICTIONARY dictionary-name password ;
```

A LOG STATUS command is not logged.

The LOG STATUS command without a DICTIONARY clause is accepted only if a dictionary is open under the authority of the dictionary Controller or of the Master Operator. If the command is accepted, a Log Status report for the dictionary is output.

The LOG STATUS command with a DICTIONARY clause is accepted only if no dictionary is currently open. It opens the log dataset of the dictionary specified by dictionary-name and checks whether the password quoted in the command is that of the dictionary Controller or the Master Operator. If it is not, the log dataset is closed and the error message:

```
DM12103E MASTER PASSWORD MISMATCH
```

is output. If the quoted password is the password of the dictionary Controller or the Master Operator, the Log Status report of the dictionary is output.

The DICTIONARY clause may be used when the dictionary is inaccessible and information regarding the roll-forward capability must be obtained.

The DICTIONARY clause is not available for use with a DIV dictionary.

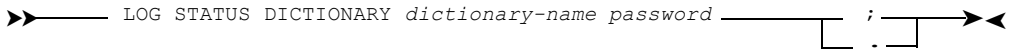
For example, you would enter:

```
LOG STATUS;
```

or

```
LOG STATUS DICTIONARY DDICT 'MSPDMR';
```

LOG STATUS Syntax



where:

dictionary-name is the name of your dictionary.

password is either the Master Operator's or the Controller's password as recorded in the dictionary.

LOG SWITCH

The LOG SWITCH command flags a currently ACTIVE (primary or secondary) area of a log dataset as AWAITING ARCHIVE, enter:

```
LOG SWITCH ;
```

or

```
LOG SWITCH UNCONDITIONAL ;
```

Note:

In this discussion, reference is made to the LOG SWITCH conditional command. *Conditional* is used for clarity only and is not a keyword. It refers to the use of the LOG SWITCH command as opposed to LOG SWITCH UNCONDITIONAL.

LOG SWITCH commands are accepted only under the authority of the dictionary Controller or the Master Operator and are not logged.

LOG SWITCH conditional is accepted only if the status of the alternate area is AVAILABLE and if the ACTIVE area has at least one transaction recorded in it.

LOG SWITCH UNCONDITIONAL is accepted whatever the status of the log dataset. However, if the alternate area is already full no further logging is possible until the log is archived.

LOG SWITCH conditional should only be used in circumstances where the use of LOG SWITCH UNCONDITIONAL might affect other users, that is, when other dictionary users are entering transactions which will be logged.

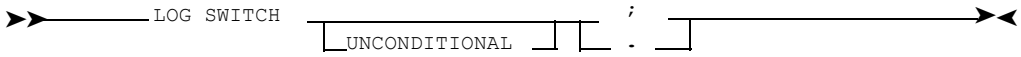
ASG recommends using the LOG SWITCH UNCONDITIONAL command when an archive is to be taken prior to an UNLOAD, to ensure that all transactions are archived. The LOG SWITCH UNCONDITIONAL command should be used when archiving is performed on a regular basis, such as with a standard procedure on an overnight basis.

A LOG SWITCH command would be used if an extensive update run required more space than is currently available on the log dataset.

In order to ensure that both areas do not remain full, LOG SWITCH UNCONDITIONAL should only be used in a procedure that includes archiving.

Although LOG SWITCH commands are not updating commands, they are not accepted if the dictionary is open in read-only mode, because update capability is needed in order to write to the log dataset.

LOG SWITCH Syntax



LOG UPDATE-COMMANDS

See "LOG ALL-COMMANDS/UPDATE COMMANDS" on page 147.

MP-AID LIST UDS-TABLES and MP-AID LIST UDS-COMPARISON-TABLES

These commands are used to list the UDS Tables, or UDS Comparison Tables, on the MP-AID.

Refer to page 158 for the syntax of the MP-AID LIST UDS-TABLES and MP-AID LIST UDS-COMPARISON-TABLES commands.

To list all of the UDS tables on the MP-AID, enter either:

```
MP-AID LIST UDS-TABLES ;
```

or

```
MP-AID LIST UDST ;
```

To list all of the UDS Comparison tables on the MP-AID, enter either:

```
MP-AID LIST UDS-COMPARISON-TABLES ;
```

or

```
MP-AID LIST UDSC ;
```

These commands can be issued by the repository Controller, a designated Controller, or the Systems Administrator. You can use all standard MP-AID LIST options (such as the DATE-ORDER keyword) with this command.

Listing Members on a Secondary MP-AID

To list UDS (Comparison) tables on a secondary MP-AID, enter either:

```
MP-AID LIST CONCATENATION mpaid-name UDS-TABLES ;
```

or

```
MP-AID LIST CONCATENATION mpaid-name UDS-COMPARISON-TABLES ;
```

where *mpaid-name* is the logical name of a secondary MP-AID.

To list all UDS tables, or all UDS Comparison tables, on the primary MP-AID and all secondary MP-AIDs, enter either:

```
MP-AID LIST ALL-CONCATENATIONS UDS-TABLES ;
```

or

```
MP-AID LIST ALL-CONCATENATIONS UDS-COMPARISON-TABLES ;
```

Description of Output

The list that is output by these commands has these column headings:

- NAME, which is the name of the UDS table or UDS Comparison table
- TYPE, identifying the table's member type (UDST or UDSC)
- DATE, the date when the member was added/replaced to the MP-AID
- TIME, the time when the member was added/replaced to the MP-AID
- BLOCKS, the number of logical blocks, if any, that are occupied by the table

and, if the list is of UDS tables:

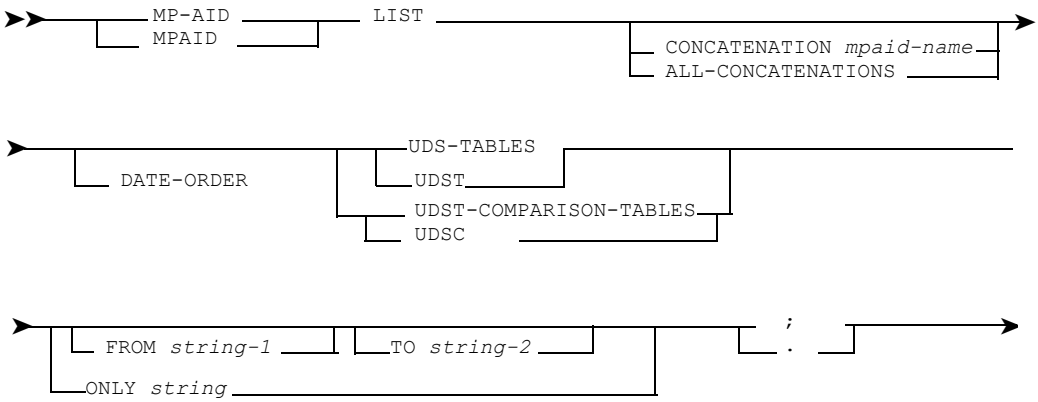
- ORIGIN, which identifies the table's origin as either a load module, which appears as LD-MOD, or user-defined (where it has been constructed onto the MP-AID)
- IN-USE-BY, which lists, where appropriate, the repositories which have this UDS table implemented

or, if the list is of UDS Comparison tables:

- COMP-NAME, the name of the UDS table with which the currently named UDS table has been compared
- RESULT, result of the comparison. If the UDS tables are compatible, COMPAT is output. If they are incompatible, ERRORS is output. If some members on the repository to which the UDS table is to be applied need re-encoding to be made compatible, ENCODE is output.

The command output first lists the primary UDST member on the MP-AID where the UDS table has been constructed onto the MP-AID. No corresponding entry is made for those UDS tables that are held as load modules. The primary UDST member is the UDS table itself. This is followed by an entry for the secondary UDST member(s) on the MP-AID. Each secondary UDST member is a record of the usage of a UDS table by a specific repository and so includes an entry under the IN-USE-BY heading both for constructed UDS tables and for UDS tables held as load modules.

MP-AID LIST Syntax



where:

mpaid-name is the logical name of a secondary MP-AID

string-1, *string-2*, and *string* specify the range of MP-AID members to be selected and can be from 1 to 10 characters.

OWNER

The OWNER command adds owner names to the dictionary, deletes owner names from the dictionary, or list the owner names held in the dictionary.

Refer to "OWNER Syntax" on page 161 for the syntax of the OWNER command.

The OWNER command is available only if the ControlManager Audit and Security facility (selectable unit CMR-DD3) is installed. This command is treated as an updating command by the automatic recovery system.

Adding Owners to the Dictionary

To add a new owner to the dictionary, enter:

```
OWNER ADD owner-name-list ;
```

where *owner-name-list* is a list of one or more owner-names made up of a string of up to 32 printable characters.

For example, to specify two new owner-names, PAYROLL and ACCOUNTS, to be added to the dictionary, enter:

```
OWNER ADD PAYROLL, ACCOUNTS ;
```

If the owner-name string is delimited (put in single or double quotes), it can include any printable character except the quotation mark. A space character is regarded as a printable character.

If two or more owner-names are listed after an ADD keyword, each except the last in the list must be followed by a comma. Entries in the list may additionally be separated by spaces.

Each owner-name declared for an ADD keyword (up to a maximum of 255) is added to the dictionary unless it is already held in the dictionary. If it is already held, the message:

owner-name ALREADY IN DICTIONARY-CANNOT BE INSERTED

is output and processing continues with the next owner-name or keyword.

Deleting Owners from the Dictionary

To delete an owner from a dictionary, enter:

OWNER DELETE *owner-name-list* ;

where *owner-name-list* is a list of one or more owner-names made up of a string of up to 32 printable characters.

If the *owner-name* string is delimited (put in single or double quotes), it can include any printable character except the quotation mark. A space character is regarded as a printable character.

If two or more owner-names are listed after a DELETE keyword, each except the last in the list must be followed by a comma. Entries in the list may additionally be separated by spaces.

Each *owner-name* declared for a DELETE keyword is deleted from the dictionary provided that:

- The *owner-name* exists in the dictionary
- No members belong to the *owner-name*
- No users are recorded as having access to members owned by *owner-name*

If the owner-name does not exist in the dictionary the message:

owner-name NOT IN DICTIONARY—CANNOT BE DELETED

is output.

If there are members belonging to the owner-name, the message:

```
owner-name STILL OWNS nnnn MEMBERS—CANNOT BE DELETED
```

where *nnnn*, the number of members still owned by this owner, is output.

If there are users recorded as having access to members owned by the owner-name, the message:

```
owner-name STILL REFERRED TO BY USERS—CANNOT BE DELETED
```

is output.

In each case, processing continues with the next owner-name or keyword.

Listing Owners

To list all the owners in a dictionary, enter:

```
OWNER LIST ;
```

If LIST is specified, then after all ADD and DELETE requirements have been processed, a list of all owner-names in the dictionary is output. The owner-names are listed in the order in which they were input to the dictionary. Associated with each owner-name is the total number of members currently recorded as having been explicitly ascribed to that owner-name in PROTECT commands.

For example, if there are three owners in the dictionary: PAYROLL, ACCOUNTS, and PERSONNEL, to specify that the owner PERSONNEL is to be deleted from the dictionary and to list the remaining owners, enter:

```
OWNER DELETE PERSONNEL LIST ;
```

This is the list format:

Owner List

Members	Owner Names
3	PAYROLL
0	ACCOUNTS

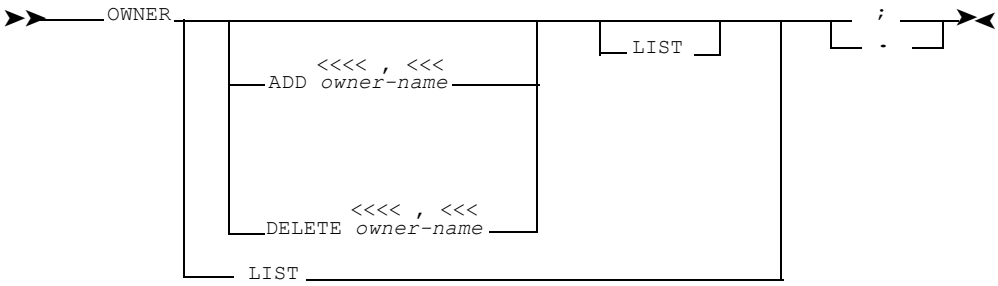
END OF OWNER LIST

If the LIST keyword is included more than once, it is actioned only once.

The Maximum Number of Owners Allowed

The maximum number of owner-names that can be recorded in the dictionary is 255. If an owner-name that, if accepted, would bring the number of owner-names recorded in the dictionary above the limit of 255 is encountered in an ADD clause, then that owner-name is rejected, and processing continues with the next element in the command. It follows that, if the number of owner-names recorded is approaching the limit, and both ADD and DELETE clauses are to be processed in an OWNER command, the DELETE clauses should be entered before the ADD clauses.

OWNER Syntax



where *owner-name* is up to 32 printable characters.

RELOAD

The RELOAD command inputs a physical copy of a dictionary from a file that was written by an UNLOAD command. A log dataset for a dictionary is established when the AND LOG clause is used.

The Controller's UNLOAD and RELOAD commands are primarily used for backup and recovery purposes. You can additionally use them to change the size of one or more BDAM or VSAM-organized dictionary datasets. However, the logical and physical block sizes of a dictionary cannot be changed by the RELOAD command itself; all block sizes in a RELOADed dictionary remain the same as those of the original dictionary.

If the input dataset was not written by an UNLOAD command, the RELOAD command is rejected and this message is output:

```
INPUT TO RELOAD IS NOT AN UNLOAD DATASET
```

The RELOAD command recreates and repopulates a dictionary from an input dataset written by an UNLOAD command. The size of the RELOADed BDAM or VSAM-organized dictionary is determined by the space allocation parameters specified in the JCL statements or VSAM cluster definitions.

The logical and physical block sizes of the index, source, and data entries datasets are determined from control information recorded in the UNLOAD input dataset.

After these three dictionary datasets have been regenerated, a new recovery dataset is created, using the physical block size as recorded in the UNLOAD input dataset.

A log dataset is established for the RELOADed dictionary only if you include AND LOG in the RELOAD command.

For a DIV dictionary, you can only specify AND LOG if the dictionary was originally VCREATEd with a log.

Refer to *ASG-Manager Products Server Facility User's Guide* for details of the RELOAD command when used for DIV dictionaries.

If a RELOAD command is rejected for any reason while running under OS, the disk space for the four or five datasets remains allocated and named by the operating system.

For a VSAM dictionary, if the control interval sizes defined for the dictionary datasets (as specified via the IDCAMS CONTROLINTERVALSIZE parameter) differ from those of the UNLOADed dictionary, the RELOAD command is rejected with the message:

```
INCOMPATIBLE CONTROLINTERVALSIZE ON dataset-name DATASET
```

The UNLOAD and RELOAD commands cannot be used to convert a dictionary from one access method to another. This can only be done by using the SAVE ALL, CREATE, VCREATE, and RESTORE ALL commands.

When you enter a RELOAD command, any dictionary that is open at that time is closed automatically before the command is actioned.

Note:

You cannot use the RELOAD command in a POST or MAIL command.

Using the RELOAD Command

To input a physical copy of a dictionary from a dataset that was previously written by an UNLOAD command, use the RELOAD command.

If logging is implemented in your dictionary and you enter:

```
RELOAD dictionary-name password ;
```

(without the optional keywords AND LOG), then after the input dataset has been written to the dictionary, the log dataset is examined to determine whether any successful updating commands were added to the log subsequent to the dictionary UNLOAD.

If any such commands were logged following the UNLOAD, the roll-forward feature is applied automatically to complete the recovery of the dictionary. This is done by reconstituting the latest updated state of the dictionary; that is, by applying all successful updating transactions from the dictionary's log.

When you RELOAD a dictionary with a log dataset, all successful updating commands added to the log subsequent to the dictionary UNLOAD are reapplied to the RELOADed dictionary, unless any result code 8 transactions are encountered.

If any result code 8 transactions were logged after the backup was taken, then dictionary roll-forward is processed only up to the first result code 8 transaction. This message is output:

```
DM12261E          ROLL-FORWARD CANNOT BE CONTINUED
                  - RESULT CODE 8 TRANSACTION REACHED
```

If this message is output, you should reapply all successful updating commands that were logged starting with the result code 8 transaction. Refer to "Recovery with Result Code 8 Transactions" on page 36 for details of how to complete dictionary recovery when result code 8 transactions are encountered.

For a VSAM dictionary with a log dataset which is not being recreated by an AND LOG clause in the RELOAD command, you must ensure that, when deleting and redefining the dictionary clusters via IDCAMS, you do not purge any existing log cluster. If you do include AND LOG in the RELOAD command to create or to recreate a log dataset for a VSAM dictionary, then the log cluster must be deleted and redefined via IDCAMS.

To change the space allocated to each BDAM or VSAM dictionary's datasets

1. Make a copy of the dictionary using the UNLOAD command.
2. Increase or decrease the space allocated to each of the dictionary's datasets, as required, either by changing the appropriate JCL statements or by redefining the datasets' space allocation via IDCAMS (depending on the type of your dictionary).
3. Rebuild the dictionary using the RELOAD command.

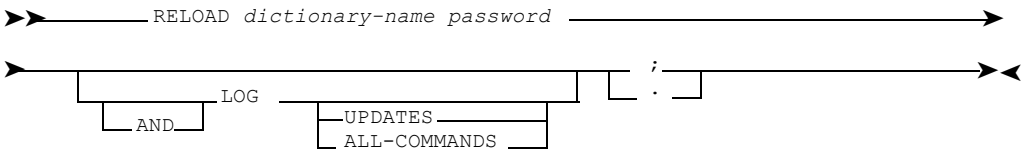
If you RELOAD a dictionary in batch, you (or the Master Operator) should examine the output messages to determine whether the reload and (if applicable) roll-forward processes have been successfully completed.

Note:

The keywords UPDATES and ALL-COMMANDS are available only if the Audit and Security facility (selectable unit CMR-DD3) is installed.

Establishing a Log Dataset

To establish a log dataset for the RELOADed dictionary, enter:



where:

dictionary-name is the name of the dictionary from which the input dataset was written

password is the Controller's password or the Master Operator's password for the UNLOADed dictionary, as recorded in the input dataset.

AND is included in the command for readability only. It has no processing significance.

The RELOAD command will continue to use the existing log dataset (if any) for the dictionary, unless you include AND LOG in the command, in which case a new log dataset is established.

You can use the AND LOG clause for these tasks:

- Set up a log dataset for the RELOADed dictionary when the UNLOADed dictionary did not previously have a log dataset.
- Set up a new log dataset for the RELOADed dictionary, when the old log dataset has become inaccessible.
- Set up a new log dataset for the RELOADed dictionary to replace an existing log dataset (for example, to enlarge the log). You must not do this if any successful updating commands have been logged between the acceptance of the UNLOAD command and the acceptance of the RELOAD command, as those commands are not reflected in the reconstituted dictionary.

For a DIV dictionary, you can only specify AND LOG if the dictionary was originally VCREATEd with a log.

Refer to *ASG-Manager Products Server Facility User's Guide* for details of the RELOAD command when used for DIV dictionaries.

Having successfully UNLOADed and RELOADed a dictionary, you should immediately issue a further UNLOAD command, in order to reestablish roll-forward capability.

If you include AND LOG in the command, but neither UPDATES nor ALL-COMMANDS, then the value of the COMTYPE parameter of the DLOG installation macro determines whether the command operates as though LOG UPDATES were stated or as though LOG ALL-COMMANDS were stated.

To log (in the log dataset of the RELOADed dictionary) all subsequent updating commands entered in the RELOADed dictionary and override the COMTYPE parameter of the DLOG installation macro, enter:

```
RELOAD dictionary-name password AND LOG UPDATES ;
```

To log (in the log dataset of the RELOADed dictionary) all subsequent commands entered in the RELOADed dictionary and override the COMTYPE parameter of the DLOG installation macro, enter:

```
RELOAD dictionary-name password AND LOG ALL-COMMANDS ;
```

Refer to the appropriate Manager Products installation publication for details of the DLOG installation macro.

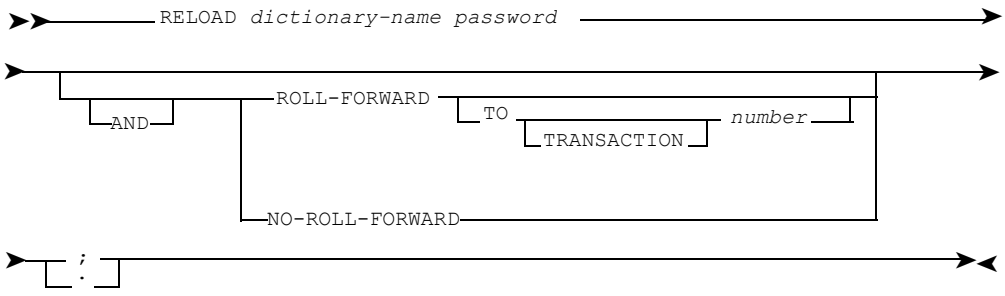
If you do *not* include AND LOG in the command and no log dataset was in use for the UNLOADed dictionary, then processing is complete when the input dataset has been written to the dictionary.

Note:

The keywords UPDATES and ALL-COMMANDS are available only if the Audit and Security facility (selectable unit CMR-DD3) is installed.

Reconstituting the Dictionary to a Particular Transaction

To reconstitute a dictionary to a particular transaction, enter:



where:

dictionary-name is the name of the dictionary from which the input file was UNLOADed.

password is the master password or the Master Operator's password as recorded for *dictionary-name.*

number is an unsigned integer ranging from one to six digits, and is the number of a transaction present in the log dataset.

AND and TRANSACTION are included in the command for readability only. They have no processing significance.

If you need to rebuild a dictionary after you have taken a backup copy of the dictionary and do not want to reconstitute the dictionary to its latest updated state, you can do one of these things:

- Prevent the roll-forward feature being activated altogether
- Use the roll-forward feature to reconstitute the dictionary to a specific transaction. (This could be useful if, for example, since the dictionary was last backed up, you used the PERFORM command with the REMOVE command and accidentally removed the wrong members, and you now want to reconstitute the dictionary to the last successful updating command entered prior to the erroneous REMOVE commands.)

To stop the automatic roll-forward, enter:

```
RELOAD dictionary-name password AND NO-ROLL-FORWARD ;
```

After this command you must enter a LOG PURGE command in order to allow the RELOADed dictionary to be opened.

To stop the automatic roll-forward at a particular transaction, enter:

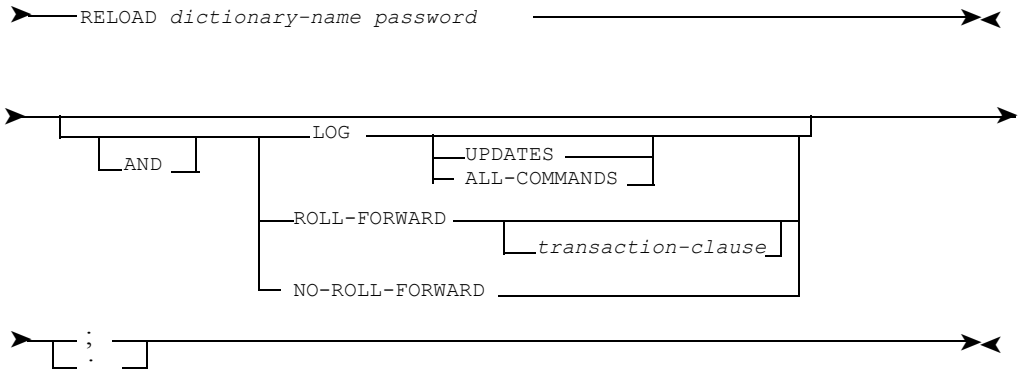
```
RELOAD dictionary-name password AND ROLL-FORWARD TO TRANSACTION n ;
```

All successful updating commands, up to and including the transaction number specified in the TO clause, are reapplied to the dictionary.

Any transactions logged after the transaction specified are not reapplied to the dictionary. You must purge these transactions using the LOG PURGE command. The dictionary cannot be reopened for subsequent processing unless all transactions added to the log subsequent to the use of an UNLOAD command are reapplied or purged.

The LOG PURGE command is not available for use with a DIV dictionary. Refer to *ASG-Manager Products Server Facility User's Guide* for details of how to drop unwanted transactions from the log.

RELOAD Syntax

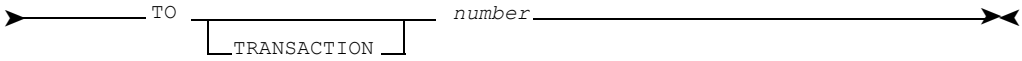


where:

dictionary-name is the name of the dictionary from which the input file was UNLOADed

password is the master password or the Master Operator's password as recorded for *dictionary-name*.

transaction-clause is:



where *number* is an unsigned integer of up to six digits, being a transaction number in dictionary-name's log dataset.

Refer to *ASG-Manager Products Server Facility User's Guide* for the syntax of the RELOAD command when used with a DIV dictionary.

REMOVE

The REMOVE command removes a member from a dictionary under the authority of the Controller's master password, even if reference is made to it by other members.

The REMOVE command does not permit a user to remove a member from a dictionary if reference is made to it by other members. When a REMOVE command is issued under the authority of the Controller's master password, this restriction does not apply. You can thus remove any member from the dictionary, regardless of whether the member is used by any other members. If the removed member is used by other members, ControlManager sets up a dummy data entries record for the removed member.

With a Multi-status Dictionary

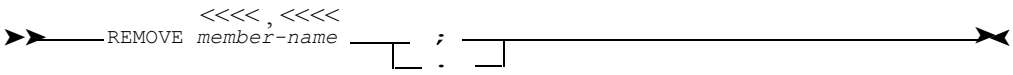
If you have the Basic Status or Advanced Status facility installed (selectable unit CMR-DD2 or CMR-AD21), the above applies in respect to a dependent current status, subject to these conditions:

- If the member has a source record but no data entries record in the current status, then it can be removed, regardless of whether it also exists in a base status; the definition in the base status will then be seen from the current status.
- If the member has a (dummy or encoded) data entries record in the current status and also exists in a base status, then it cannot be REVERTed.
- If a member that exists in a base status is no longer required in the dependent current status, then it can be converted into an OBSOLETE-DEFINITION in the current status.

The above rules apply whatever the member-type of the member in the base status (so that they apply also if the member is an OBSOLETE-DEFINITION in the base status).

For details of the REMOVE command as available to all users, please refer to *ASG-Manager Products Dictionary/Repository User's Guide*.

REMOVE Syntax



where *member-name* is the name of a dictionary member.

RESERVE

The Executive Command RESERVE is used to define the beginning of a Logical Unit of Work (LUW). The additional keyword SPEED is available for Controllers and Systems Administrators. See "SPEED Keyword in the RESERVE Command" on page 169.

An LUW is a group of commands that are treated as one command for the purposes of processing and can be treated as one command for recovery purposes. You can define an LUW for the dictionary or the MP-AID.

To define the beginning of an LUW for the dictionary, enter:

```
RESERVE DICTIONARY mode ;
```

To define the beginning of an LUW for the MP-AID, enter:

```
RESERVE MP-AID mode ;
```

where *mode* is UPDATE if the LUW contains update or a combination of update and interrogation commands or READ-ONLY if the LUW contains interrogation commands only. Updates are not allowed in READ-ONLY LUWs.

Synonyms for these keywords are EXCLUSIVE and SHARED respectively. To terminate an LUW, use the RELINQUISH command.

If you define a dictionary LUW in UPDATE/EXCLUSIVE mode, you may choose to commit each update as it completes. Committing an update causes a permanent change to the dictionary (or MP-AID for an MP-AID LUW). To commit each update as it completes, enter:

```
RESERVE DICTIONARY UPDATE COMMIT ;
```

or

```
RESERVE DICTIONARY UPDATE ;
```

If you do not specify a keyword after the mode, the default setting is COMMIT. If any abnormal termination occurs during the processing of a dictionary update, the dictionary is automatically recovered to the state it was in before the update began.

Alternatively, you may commit all updates together when the LUW is complete by beginning the LUW:

```
RESERVE DICTIONARY UPDATE ROLLBACK ;
```

and specifying COMMIT in the RELINQUISH command when you terminate the LUW. If any abnormal termination occurs, the dictionary is recovered to the state it was in before the LUW began executing.

You cannot specify ROLLBACK when you terminate the LUW if COMMIT is specified (or allocated by default) at the beginning of the LUW, since updates are committed as they complete and cannot therefore be rolled back.

The COMMIT and ROLLBACK keywords are not available in READ-ONLY/SHARED mode or for an MP-AID LUW.

SPEED Keyword in the RESERVE Command

The SPEED keyword optimizes processing by minimizing the number of physical I/Os for processing commands within a Logical Unit of Work (LUW). However, when the SPEED keyword is specified, there is no recovery capability in the event of abnormal termination. It is therefore restricted for use by Controllers and Systems Administrators.

The SPEED keyword is accepted, but has no effect, when used as part of a DIV dictionary LUW definition.

To optimize processing a dictionary LUW, define the LUW by entering:

```
RESERVE DICTIONARY UPDATE SPEED ;
```

To optimize processing an MP-AID LUW, define the LUW by entering:

```
RESERVE MP-AID UPDATE SPEED ;
```

Fast processing is achieved because dataset physical I/O occurs only when absolutely necessary; that is, updates carried out while the LUW is active are not committed to the dictionary or MP-AID unless the buffer pools become full and more data needs to be read in. In this case, an updated buffer must be written out to disk to allow new data to be read in.

Using the SPEED capability can therefore reduce processing requirements considerably. For example, by increasing the dictionary buffer pools and using the SPEED capability, you can reduce elapsed time and the number of I/Os for a RESTORE ALL command by 70 to 80 percent and CPU usage by 30 percent. Maximum savings can be achieved by further increasing the dictionary buffer pools.

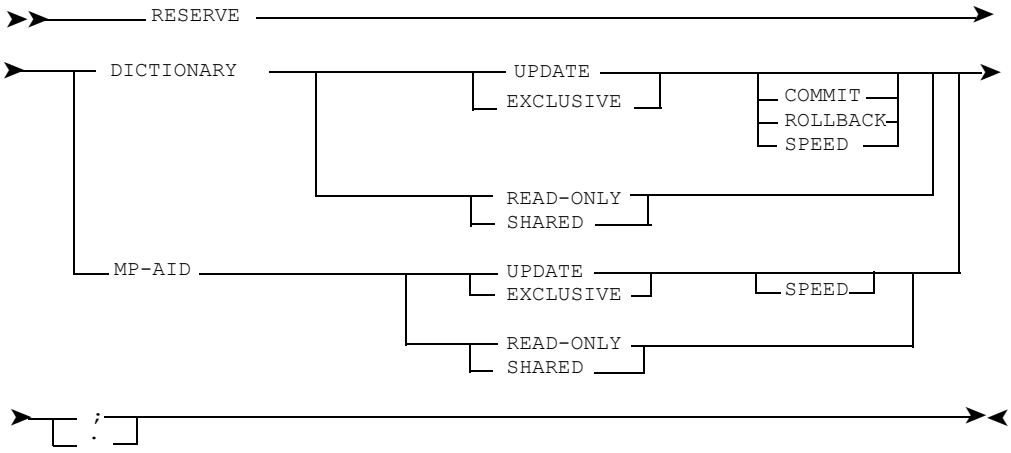
It is possible to almost eliminate physical I/Os if the buffer pool for each dataset is large enough to hold enough data to satisfy all the requirements of the commands within an LUW. Some I/Os may still take place on the Recovery dataset in order to maintain the integrity of the dictionary; for example, in situations where an update command fails and any updated members need to be recovered to their previous state.

If logging is active, a small number of I/Os will occur on the Log dataset for each command to be logged. However, since the SPEED capability minimizes the number of physical I/Os for processing commands within an LUW, the number of physical I/Os recorded on the log for such transactions may not be representative of the number of I/Os that would have occurred without the SPEED capability being active.

When using the SPEED keyword, you should be aware that if abnormal termination occurs, the dictionary or MP-AID can only be recovered from a backup copy. You should therefore only use the SPEED capability in circumstances where recovery from backup is acceptable or no data is lost. For example, you may consider issuing a RESTORE ALL into an empty dictionary or multiple CONSTRUCTs into an empty MP-AID since, in the event of any abnormal termination, you need only to recreate the dictionary or MP-AID datasets and reexecute the failed LUW.

To terminate an LUW, use the RELINQUISH command. The RELINQUISH command is available to all users.

RESERVE Syntax



Note:

The SPEED keyword is available for use by Controllers and Systems Administrators only.

RESTORE

The RESTORE command restores all or selected parts of a repository from a dataset written by a SAVE command. It can also be used to reorganize a repository. Refer to *ASG-Manager Products Performance Tuning* for details of reorganizing your repository.

Refer to "RESTORE Syntax" on page 177 for the syntax of the RESTORE command.

You can restore everything from the input dataset, provided that the dataset was created using SAVE ALL.

Alternatively, you can restore these items:

- Security information
- Status information
- All source records
- Specific source records

If you enter:

```
RESTORE ;
```

then:

- The keyword ALL is assumed if the input dataset was written by a SAVE ALL command.
- The keyword SOURCE is assumed if the input dataset was written by a SAVE command.

The commands:

```
RESTORE SOURCE ;
RESTORE MEMBERS member-name-list ;
```

can be used in conjunction with a SAVE ALL or a SAVE selection command. Variants of the RESTORE command with these keywords:

- ALL
- SECURITY
- STATUS
- FROM-STATUS

can only be used in conjunction with a SAVE ALL command.

If the input dataset was not written by a SAVE ALL command, the RESTORE command is rejected.

With the RESTORE SOURCE and RESTORE MEMBERS commands, each restore of a source member is treated as a separate updating command by the automatic error recovery system.

Restoring Everything

To restore all index-names, enter:

```
RESTORE ALL ;
```

The whole information content of the input dataset is written to the current repository (the receiving repository), provided that:

- The receiving repository is empty, that is, only CREATE, DICTIONARY, and AUTHORITY commands have previously been issued. If the receiving repository is not empty, the RESTORE ALL command is rejected.
- The receiving repository has at least as many statuses as there are in the saved repository. If there are insufficient statuses in the receiving repository, the RESTORE ALL command is rejected.

The information restored by RESTORE ALL includes, in addition to the members' source and data entries records these items:

- Password/owner information
- Status information
- Alias-type keyword list
- Data entries records of aliases and catalog classifications and user defined indexed attributes
- Date and time information of source and data entries records
- Expiry dates and times of any members that were locked at the time the SAVE ALL command was entered
- Members protection information

If the User Defined syntax facility is installed, a RESTORE ALL command automatically creates a secondary UDS-TABLE member in the MP-AID recording the UDS table usage of the repository.

Restarting RESTORE ALL

A RESTORE ALL command is executed in two stages.

In the first stage, all source records are restored and skeleton data entries records are written. If the command fails during the first stage you must create a new repository before you reissue the RESTORE ALL command. You must also correct the error that caused the RESTORE command to fail.

In the second stage, checkpointing begins and all members are re-encoded in a series of different passes. When a member fails to encode, checkpointing is switched off until the next pass. All successful re-encodes are marked as re-encoded.

If the processing of a RESTORE ALL command fails during the second stage, you can reopen the repository and reenter the RESTORE ALL command. Processing restarts from the last checkpoint and skips over the successfully re-encoded members.

For a DIV dictionary checkpointing is disabled and a failed RESTORE ALL command cannot be restarted.

It is possible that, when you restore a member, restored aliases will not conform with current validation rules. However, warning messages only will be generated. For further information on alias validation, refer to "CONTROL NEW-ALIASES" on page 117.

Restoring Security Information

Security information consists of password/owner information for the whole repository and protection information for each member.

You can restore password/owner information, then restore protection information for members in later commands. If only one status is to be restored you can restore all security information in one command.

To restore only password/owner information, enter:

```
RESTORE SECURITY ;
```

Password/owner information from the input dataset is restored into the receiving repository, provided that no password/owner information is already present. If the information is already present, the RESTORE command is rejected.

Any subsequent RESTORE command that specifies SECURITY checks that the password/owners on the repository match those on the SAVE dataset. If the details match protection information is restored with the members. If they do not match the RESTORE command is rejected.

To restore both password/owner and protection information to the current status of the receiving repository, enter:

```
RESTORE SECURITY SOURCE ;
```

Restoring Status Information

To restore status information, enter:

```
RESTORE STATUS ;
```

Status names from the input dataset are restored into the receiving repository. If the repository contains status information, the RESTORE command is rejected.

Restoring Alias Information

To restore alias information, enter:

```
RESTORE ALIAS ;
```

Alias definitions from the input dataset are restored into the receiving repository.

If the repository contains any member definitions, the RESTORE command is rejected.

Restoring Source Records or Specific Members

To restore source records, enter:

```
RESTORE SOURCE ;
```

To restore the source records of specific members, enter:

```
RESTORE MEMBERS member-name-list ;
```

where *member-name-list* is a list of member names, each separated by a comma.

If the input dataset was written by a SAVE selection command, it contains source records from one status only; source records from that status are restored into the current status of the receiving repository.

If the input dataset was written by a SAVE ALL command, then the source records are taken from the earliest frozen status of the saved dataset, or, if there are no frozen statuses, from the first named status.

If you include the FROM-STATUS option, the source records are taken from the status named in that clause.

If you include the REPLACE option, any source record being restored can replace the source record of a member of the same name in the receiving repository, provided that the member in the repository is not locked by any user other than the Controller.

The source record of a member is not replaced if you omit the REPLACE option or if a member of the same name already exists in the current status of the receiving repository.

Processing continues with the next member or command.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of member locking.

Suppressing Output

By default a list of the names of the restored members is output during the execution of a RESTORE command.

After a RESTORE ALL command, aliases, catalog classifications, and indexed attributes of encoded members are also included.

To suppress the list of restored members during a RESTORE command, include the keyword NO-PRINT or NOPRINT in your command, for example:

```
RESTORE SOURCE NO-PRINT ;
```

Examples

To restore and reorganize a repository, enter:

```
RESTORE ALL ;
```

The receiving repository must be empty.

To add the saved source of members to the current status of the receiving repository, provided the members do not already exist in the repository, enter:

```
RESTORE SOURCE ;
```

The members' ownership and protection details are not restored.

To replace the source records of four members in the current status with the source records of these members from the input dataset, enter:

```
RESTORE MEMBERS DEPARTMENT, JOB-TITLE, SALARY, EMPLOYEE-NUMBER
REPLACE ;
```

To restore all password/owner information from the input dataset, enter:

```
RESTORE SECURITY ;
```

To restore all status names from the input dataset as non-frozen statuses, enter:

```
RESTORE STATUS ;
```

To restore all password/owner information and the source records of the members PROD and PROD-RPT from the status DEV1, with their protection details (if any), enter:

```
RESTORE SECURITY MEMBERS PROD-RPT. PROD FROM-STATUS DEV1 ;
```

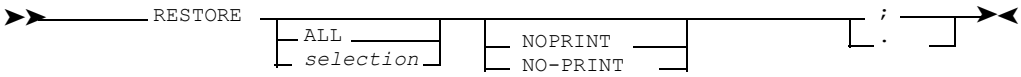
To restore all source records and status names from the input dataset and suppress the list of the members restored, enter:

```
RESTORE STATUS SOURCE NO-PRINT ;
```

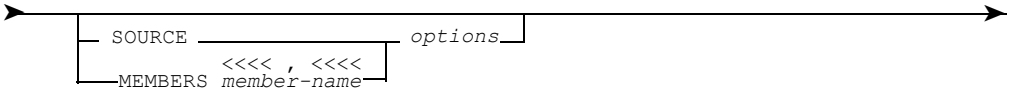
To restore all source records from status RELEASE on the input dataset, enter:

```
RESTORE SOURCE FROM-STATUS RELEASE ;
```

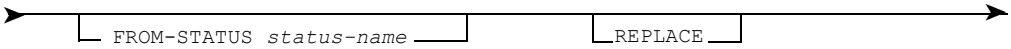
RESTORE Syntax



where *selection* is:



where *options* is:



where:

status-name is a status in the input dataset.

member-name is the name of a repository member.

ROLL-FORWARD

The ROLL-FORWARD command can complete the recovery of a BDAM or VSAM-organized dictionary by applying updating transactions from its log after the dictionary has been reconstituted by non-Manager Products software (instead of by a RELOAD command), or if an interruption has occurred during the roll-forward phase of a RELOAD command.

A DIV dictionary cannot be reconstituted by non-Manager Products software. For further information refer to "Dictionary Backup" on page 44.

To ROLL-FORWARD to a specific transaction, enter:

```
ROLL-FORWARD dictionary-name password TO TRANSACTION number ;
```

where:

password is the master password as recorded in the dictionary.

number is the number of a transaction recorded in the dictionary's log.

The keyword TRANSACTION has no processing significance; it is available in the command only for readability.

The ROLL-FORWARD operation is normally performed using the most recent backup copy of the dictionary. If an earlier version is to be used, then all subsequent archives must be available.

The ROLL-FORWARD command first opens the dictionary specified by dictionary-name and checks whether the password quoted in the command is that of the dictionary Controller or the Master Operator. If it is not, the dictionary is closed and the error message:

```
DM12103E MASTER PASSWORD MISMATCH
```

is output. If the quoted password is the password of the dictionary Controller or the Master Operator, the roll-forward is processed.

The ROLL-FORWARD command may be used with the RELOAD command, or alone if the dictionary has been reconstituted by non-Manager Products software. It completes the recovery of a dictionary by applying updating transactions from the dictionary's log. All successful updating transactions logged since the non-Manager Products backup was taken, or since the transaction at which an interruption during the roll-forward phase of a RELOAD occurred, are reapplied to the dictionary, unless the TO clause is present in the command or result code 8 messages are recorded.

If, when the TO clause is specified in the command, no result code 8 messages are encountered, all successful updating transactions logged since the non-Manager Products backup was taken, or since the transaction at which an interruption during RELOAD occurred, up to and including the transaction numbered as specified in the TO clause, are reapplied to the dictionary.

Any transactions logged after the transaction specified are not reapplied to the dictionary. These transactions must be purged using the LOG PURGE command. The dictionary cannot be reopened for subsequent processing unless all logged transactions since the UNLOAD are reapplied or purged.

The LOG PURGE command cannot be used with a DIV dictionary. Refer to *ASG-Manager Products Server Facility User's Guide* for details of how to drop unwanted transactions from the log.

If any result code 8 transactions were logged since the non-Manager Products backup was taken, the ROLL-FORWARD command is processed only up to the first result code 8 transaction. An error message is output:

```
DM12261E ROLL-FORWARD CANNOT BE CONTINUED
- RESULT CODE 8 TRANSACTION REACHED
```

If this message is output, the dictionary Controller should reapply all successful updating commands that were logged starting with the result code 8 transaction. Extra care should be taken in this procedure.

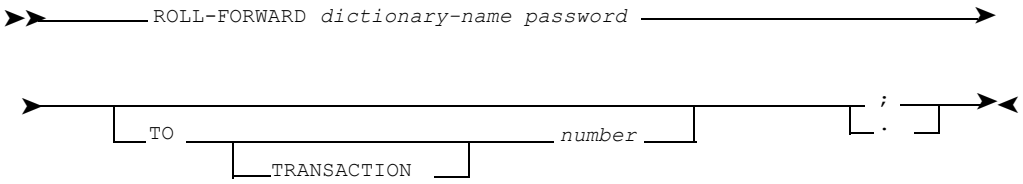
After the ROLL-FORWARD command has been processed, the Controller or Master Operator should examine the output messages to determine whether the roll-forward process has been successfully completed.

See the RELOAD command in "Dictionary Backup" on page 44 for details of how to RELOAD with no roll forward.

For example, you would enter:

```
ROLL-FORWARD DDICT 'MSPDMR';
ROLL-FORWARD DDICT MASTEROP TO 4387 ;
```

ROLL-FORWARD Syntax



where:

dictionary-name is the name of your dictionary.

password is either the Master Operator's password or the master password as recorded in the dictionary.

number is the number of a transaction recorded in the dictionary's log.

SAVE

You can save everything in a dictionary by using the keyword ALL after the command identifier. Alternatively, you can refine the SAVE command by including one or more selections and/or one or more secondary selections in the command.

Refer to "SAVE Syntax" on page 188 for the syntax of the SAVE command.

If you include selection in the SAVE command, you can select what you want to save:

- KEPT-DATA: members kept in a KEPT-DATA list
- LOCKED: currently locked members
- MEMBERS *member-name-list*: a list of specific member names
- DUMMIES: dummy members
- SOURCES: source records

(If neither ALL nor one of the above selections is stated in the command, the selection SOURCES is assumed.)

Additionally, you can further refine the operation of the command by including a secondary selection in any variant of the SAVE command:

- By name
- By status
- By time
- By user

You can exclude particular types of member/index-name from being saved by including an EXCEPT clause in the command.

If you do not require output from the SAVE command to be generated, you can suppress the output by using NO-PRINT in the command.

If ALL is not specified in the SAVE command, only source records are saved. Members are viewed from the current status and are saved as relevant to that status.

Note:

Details relating to locking are available only if the Workstation interface facility (selectable unit CMR-WS01) is installed.

Details relating to statuses are available only if either the Basic or Advanced Status facility (selectable units CMR-DD2 or CMR-AD2) are installed.

Details relating to members' protection are available only if the Audit and Security facility (selectable unit CMR-DD3) is installed.

User-defined indexed attributes are only available if the User Defined Syntax facility (selectable unit CMR-UD1) is installed.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for a definition of member/index-names.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of secondary selections.

Saving Everything in a Dictionary

To save everything in a dictionary, enter:

```
SAVE ALL;
```

The SAVE ALL command specifies that all of the dictionary is to be written to the output dataset. This is the only form of the SAVE command that, as well as saving members' source records, saves all of these items:

- Members' data entries records
- Members in all statuses
- The data entries records of aliases and catalog classifications
- Data entries records of user defined indexed attributes
- Members' protection information
- The expiration dates and times of any members that are locked
- Security information
- Status information
- The alias-type keyword list
- The UDS specification

It is recommended that you DIAGNOSE the dictionary to be saved prior to entering the SAVE command.

Saving Source Records, Dummy, or Real Members

To save the source records of all member/index-names with a source record, irrespective of whether they also have data entries records, enter:

```
SAVE SOURCES ;
```

To save all dummy members, enter:

```
SAVE DUMMIES ;
```

You can refine the SAVE DUMMIES variant of the command to exclude specific types of member/index-names from being saved by using EXCEPT.

For example, the command:

```
SAVE DUMMIES ONLY ACC EXCEPT FILES, GROUPS ;
```

saves all dummy members whose names start with the character string ACC, excluding those member/index-names of the types FILE or GROUP.

To save all non-dummy members, that is only those members that have been defined and encoded, enter:

```
SAVE REAL ;
```

You can refine the SAVE REAL variant of the command by using the EXCEPT keyword to exclude specific types of member/index-names from being saved. For example, the command:

```
SAVE REAL EXCEPT PROGRAMS, MODULES ;
```

saves all defined and encoded members except PROGRAMs and MODULEs.

ASG recommends that you DIAGNOSE the dictionary to be saved prior to entering the SAVE command.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of member types.

Saving the Members in a KEPT-DATA List, or Locked Members or Particular Members

To save member/index-names in an unnamed KEPT-DATA list, enter:

```
SAVE KEPT-DATA;
```

To save member/index-names in a named KEPT-DATA list, enter:

```
SAVE KEPT-DATA IN list-name ;
```

where *list-name* identifies a particular named KEPT-DATA list. To save only those members which are locked, enter:

```
SAVE LOCKED ;
```

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of locking.

To save one or more named members, enter:

```
SAVE MEMBERS member-name-list ;
```

where *member-name-list* is one or more member names separated by commas.

Member/index-names in a KEPT-DATA list are processed in the order in which they appear in the KEPT-DATA list. Members in a list of member names are processed in the order they are specified in the member-name-list.

If you specify the keyword ALPHABETICALLY in the command, the member/index-names are processed in alphanumerical order.

For example:

```
SAVE MEMBERS EMP-CODE  ACC-RPT, ACC-CODE ALPHABETICALLY ;
```

processes the member ACC-CODE first, then ACC-RPT followed by EMP-CODE. Locked members are automatically processed in alphanumerical order.

If you specify KEPT-DATA and/or LOCKED and/or MEMBERS *member-name-list* in the command, then all member/index-names that fall within any of the selection criteria included in the command are saved. Member/index names that appear in more than one of the categories are processed only once.

For example:

```
SAVE KEPT-DATA IN ADDRESS MEMBERS EMP-LIST, ACC-TOT ;
```

saves the member/index-names in the KEPT-DATA list named ADDRESS followed by the members EMP-LIST and ACC-TOT.

You can refine the above variants of the SAVE command to exclude specific types of member/index-name from being saved by using EXCEPT.

For example, the command:

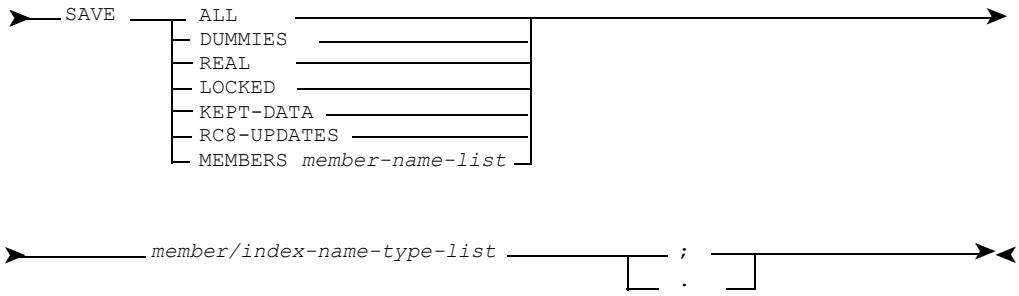
```
SAVE KEPT-DATA EXCEPT ITEMS ;
```

saves all member/index-names held in the KEPT-DATA list, excluding those of the type ITEM.

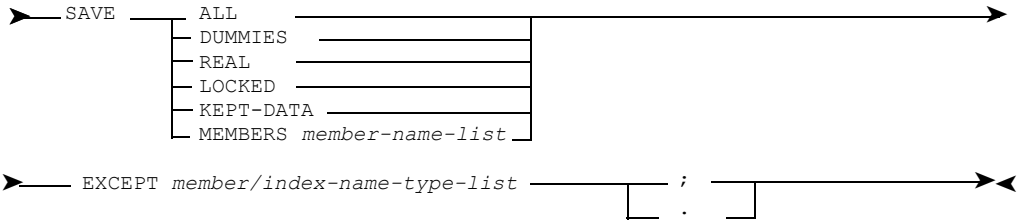
It is recommended that you DIAGNOSE the dictionary to be saved prior to entering the SAVE command.

Tailoring a SAVE Command to Include or Exclude Specific Types of Member/index-name

To save specified types of member/index-names, enter:



To save all member/index-names, except those of a specified type, enter:



where *member/index-name-type* is the interrogate keyword for any member type, or any collective member type, available in your dictionary, or the keywords ALIASES, CATALOGS, and (if the User Defined Syntax facility, selectable unit CMR-UD1, is installed) ATTRIBUTES. When more than one member/index-name-type is specified, each must be separated by a comma.

Use the SHOW UDS command to find out which member types are available in your dictionary.

For example, to save only locked members of the types FILE and ITEM, enter:

```
SAVE LOCKED FILES, ITEMS ;
```

It is recommended that you DIAGNOSE the dictionary to be saved prior to entering the SAVE command.

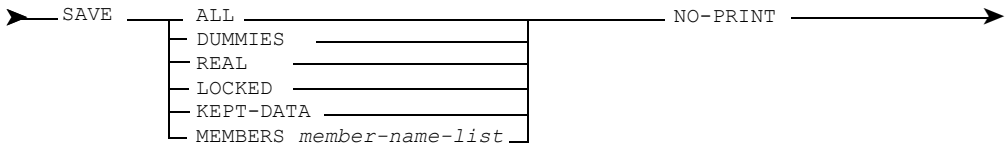
Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for a definition of member/index-names and member types.

Selecting What You Want to Process by Status, Name, Time, or User

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for details of secondary selections.

Suppressing the Output from a SAVE Command

To suppress the output of a list of which member/index-names have been processed during a SAVE command, enter:



For example, the command:

```
SAVE ALL NO-PRINT ;
```

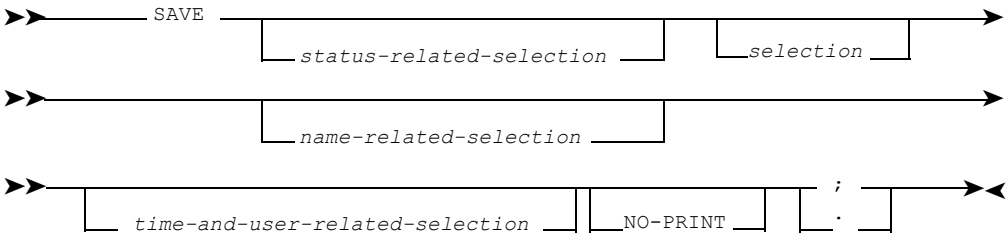
outputs all member/index-names in the dictionary, in a partially reorganized form, to the tape specified in the job control statements. No output from the SAVE command is printed.

You can spell this keyword two ways: either NO-PRINT or NOPRINT.

It is recommended that you DIAGNOSE the dictionary to be saved prior to entering the SAVE command.

Processing Complex SAVE Commands

If you want to process complex SAVE commands, you can include one or more selections and/or one or more secondary selections, in this format:



Several status-related-selection criteria and/or several time-and-user-related-selection criteria can be combined in the SAVE command (together with any selection criteria and a NO-PRINT clause) to process complex SAVE commands.

For example, to save all locked member names which are NEW or are UNVERIFIED, enter:

```
SAVE NEW, UNVERIFIED LOCKED ;
```

If, subsequently, you want to refine the operation of the SAVE command to process only those members whose names include the character string EMP, which were entered in the dictionary after the 1st January 1999, by the user (or department) whose password for the AUTHORITY command is PERSONNEL, enter:

```
SAVE NEW, UNVERIFIED LOCKED WHEN ANY EQ EMP IF INTRODUCED BY
PERSONNEL AFTER '1 JAN 99' ;
```

Having decided which member/index-names you want to save, you have the option of suppressing the output of the SAVE command by using the NO-PRINT clause, as follows:

```
SAVE NEW, UNVERIFIED LOCKED WHEN ANY EQ EMP IF INTRODUCED BY
PERSONNEL AFTER '1 JAN 99' NO-PRINT ;
```

Example

```
SAVE ALL ;
```

saves everything in a dictionary.

```
SAVE NEW, UNVERIFIED WHEN ANY EQ PROD IF NOT AMENDED BY PRODUCTION
BETWEEN '1 MAY 1999' AND '31 MAY 1999' ;
```

saves all source records which exist only in the current status or member/index-names with unencoded source records (or differing source and encoded records), which contain the character string PROD in their names, that were not updated by the user (or department) named PRODUCTION during the month of May 1999.

```
SAVE LOCKED NO-PRINT ;
```

saves all locked members, but does not output details of the saved members.

```
SAVE CHANGED KEPT-DATA EXCEPT FILES ;
```

saves all changed member/index-names in the KEPT-DATA list, which exist in the current status and any base status, excluding those of the type FILE.

```
SAVE SOURCES IF NOT INTRODUCED BY MASTER ;
```

saves the source records of all member/index-names not entered into the dictionary by the Controller.

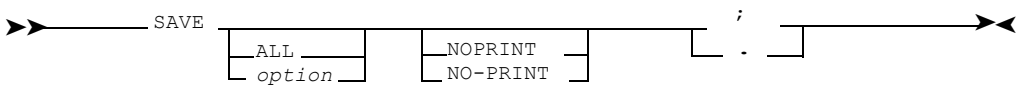
```
SAVE MEMBERS PROD-RPT, PROD, SYSTEMB2, SYSTEMB ALPHABETICALLY ;
```

saves the four members stated in the list of member names in the order PROD, PRODRPT, SYSTEMB, and SYSTEMB2.

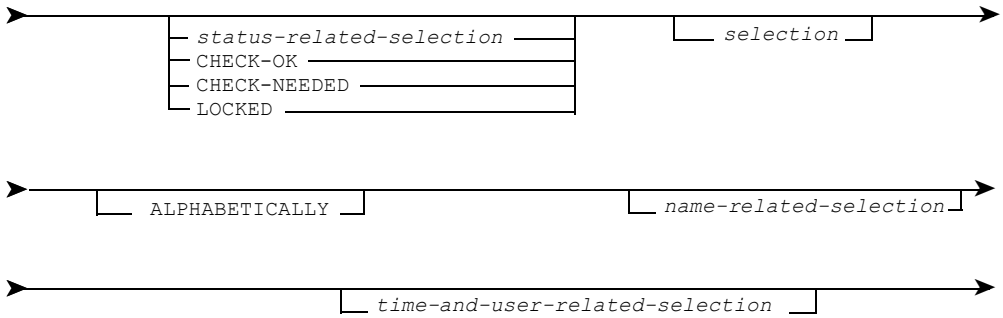
```
SAVE EXCEPT LOGON-PROFILES, GLOBAL-PROFILES, PROFILES ;
```

saves all source members except those of the types LOGON-PROFILE, GLOBAL-PROFILE, and PROFILE.

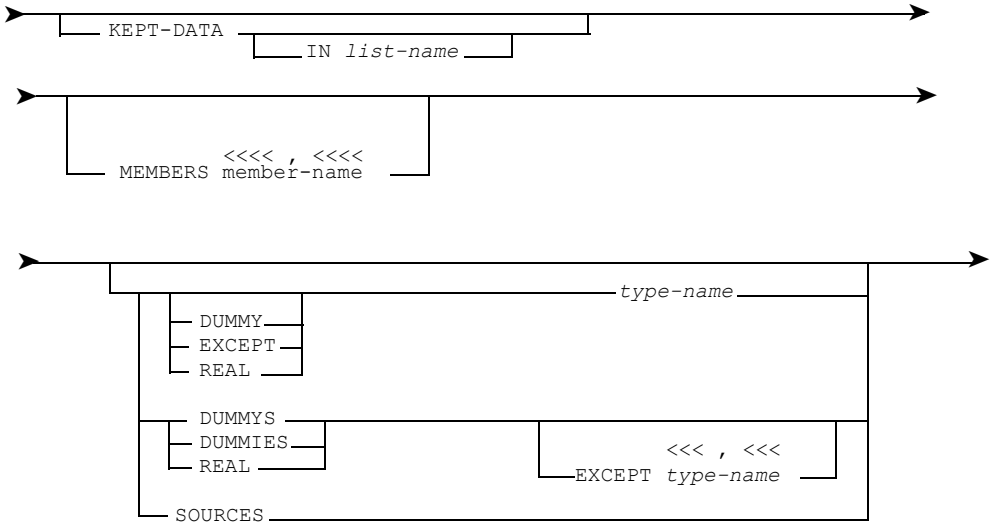
SAVE Syntax



where *option* is:



where *selection* is:

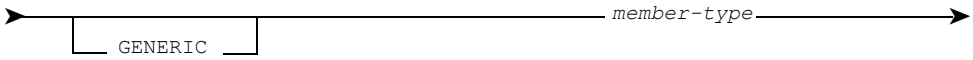


where:

list-name is the name of a KEPT-DATA list

member-name is the name of a repository member

type-name is:



where *member-type* is the interrogate keyword for any member type, or any collective member type, available in your repository.

SECURITY

The SECURITY command is used to insert user names with associated passwords into the dictionary, to delete user names and associated passwords from the dictionary, and to list the user names and associated passwords held in the dictionary.

When the Audit and Security Facility is installed the SECURITY command has additional functions:

- To establish security levels which must be satisfied before any updating commands will be accepted from a user
- To insert user names with associated security information into the dictionary
- To delete user names and associated security information from the dictionary
- To list the user names and associated security information held in the dictionary

Refer to "SECURITY Syntax" on page 200 for the syntax of the SECURITY command.

The SECURITY command is used to insert, alter, or delete user names and associated passwords from the dictionary; and/or to set up security levels for owners; and/or to establish lock periods; and/or to list the user names and associated passwords held in the dictionary.

There are three versions of the SECURITY command:

- The nucleus version, which is used to insert, alter, delete, or list user security records
- An extended version, which is used to insert, alter, delete, or list owner security records and to establish insert and protect levels in a dictionary
- An extended version, which is used to establish the period of locks for a dictionary or for individual users and to list locked members in output from the SECURITY LIST GIVING LOCKS command

Before the SECURITY command can be entered, the dictionary must already have been created, opened, and accessed using the CREATE, DICTIONARY, and AUTHORITY (quoting the Controller's password) commands.

The SECURITY command must be qualified with one, or more of these keywords:

- IDENTITY: inserting, altering and deleting users' security records
- OWNER: establishing owner security levels
- INSERT Level: establishing insert levels in a dictionary
- PROTECT Level: establishing protect levels in a dictionary
- RETENTION-PERIOD: establishing the period of locks for a dictionary
- IDENTITY RETENTION-PERIOD: establishing the period of locks for an individual user
- LIST: listing security information

The SECURITY command can contain any number of IDENTITY clauses up to a maximum of 65535 and/or a LIST clause. If both the IDENTITY and the LIST clauses are present, they can be in any order.

The SECURITY command is treated as an updating command by the automatic recovery system.

Note:

Details relating to locking are available only if the Workstation Interface facility (selectable unit CMR-WS01) is installed.

Details relating to members' protection are available only if the Audit and Security facility (selectable unit CMR-DD3) is installed.

Inserting, Altering, and Deleting User Security Records

The SECURITY IDENTITY variant of the SECURITY command can complete these tasks:

- Insert a user security record
- Alter a user security record
- Delete a user security record

Inserting User Security Records

To insert new user security information into a dictionary, enter:

```
SECURITY IDENTIFY user-name <level> password ;
```

where:

user-name is a string of up to 32 printable characters, identifying a user. If user-name is delimited (put in single or double quotes), it can include any printable character except the quotation mark. A space character is regarded as a printable character.

level is the general security level of the user who is identified by user name, being an integer in the range 0 to 254.

password is a character string of up to eight printable or non-printable characters, being the password that must be quoted in AUTHORITY commands by the user who is identified by user name.

Level establishes a user's security level. Whenever the level is omitted, a default level of one is assumed. A non-zero value is meaningful only if the ControlManager Audit and Security facility (selectable unit CMR-DD3) is installed.

If the Audit and Security facility is not installed, a non-zero value for level is accepted for compatibility but has no effect.

If you want to insert security information about a new user, you should use the SECURITY command with an IDENTITY clause specifying:

- A user-name not previously recorded in the dictionary
- A password not previously recorded in the dictionary
- A non-zero general security level is entered

For example, to establish three users of the dictionary (with the password of the first user, Johnson, including non-printable characters), enter:

```
SECURITY IDENTITY 'JOHNSON  'PMGR   '
IDENTITY 'ALLEN'  'PRO1'
IDENTITY 'SMITH.A'  'PRO2'.
```

Altering User Security Records

To alter a user's security information, enter:

```
SECURITY IDENTITY user-name <level> password ;
```

where:

user-name is a string of up to 32 printable characters, identifying a user. If user-name is delimited (put in single or double quotes), it can include any printable character except the quotation mark. A space character is regarded as a printable character.

level is the general security level of the user who is identified by user name, being an integer in the range 0 to 254.

password is a character string of up to eight printable or non-printable characters, being the password that must be quoted in AUTHORITY commands by the user who is identified by user name.

Level establishes a user's security level. A non-zero value is meaningful only if the ControlManager Audit and Security facility (selectable unit CMR-DD3) is installed. If you do not include a level in the SECURITY command, when the Audit and Security facility is installed, a default level of one is assumed.

If the Audit and Security facility is not installed, a non-zero value for level is accepted, for compatibility, but has no effect.

If you want to alter an existing user's security information, you should use the SECURITY command with an IDENTITY clause specifying:

- A user name already recorded in the dictionary, and
- The password recorded against that user name, and
- A non-zero general security level

The information in the IDENTITY clause replaces the previously recorded security information of the specified user. To change a user's password, the user's record must first be deleted, and then re-inserted with the new password specification.

Deleting User Security Records

To delete a user's security information, enter:

```
SECURITY IDENTITY user-name <level> password ;
```

where:

user-name is a string of up to 32 printable characters, identifying a user. If the user name is delimited (put in single or double quotes), it can include any printable character except the quotation mark. A space character is regarded as a printable character.

level is the general security level of the user who is identified by user name, being an integer in the range 0 to 254.

password is a character string of up to eight printable or non-printable characters, being the password that must be quoted in AUTHORITY commands by the user who is identified by user name.

Level establishes a user's security level. If the level is omitted, a default level of one is assumed. If the Audit and Security facility is not installed, a non-zero value for level is accepted for compatibility but has no effect. If the Audit and Security facility (selectable unit CMR-DD3) is installed, a non-zero value is meaningful.

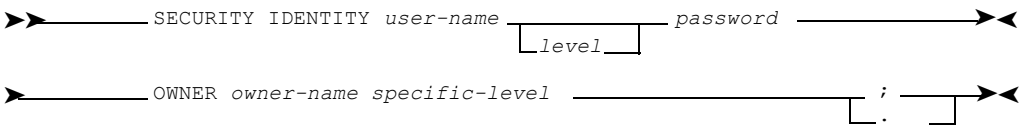
If you want to delete a use's security record, you should use the SECURITY command with an IDENTITY clause specifying a user-name already recorded in the dictionary, the password recorded against that user-name, and a level of zero.

If a user's record is deleted without first removing its OWNER clauses, it will not be possible subsequently to delete the owner name from the dictionary (should this become necessary) without first re-instating the user and then going through the above procedure.

If the ControlManager Audit and Security facility is installed (selectable unit CMR-DD3), you should use two successive SECURITY commands to delete the record of a user who has access to owned members. The first one should alter the user information by omitting any OWNER subordinate clauses from the IDENTITY clause; the second one should delete the user information.

Establishing Security Levels for Owners

To establish security details for an owner in a dictionary, enter:



where:

user-name is a string of up to 32 printable characters, identifying a user.

level is the general security level of the user who is identified by user name, being an integer in the range 0 to 254. The general security level is the user's security level when dealing with non-owned members.

The general security level must be a non-zero integer unless you are deleting a user's security record from a dictionary. If level is omitted, a default level of one is assumed.

password is a character string of up to eight printable or non-printable characters, being the password that must be quoted in AUTHORITY commands by the user who is identified by user-name.

owner-name is the name of an owner as previously recorded in the dictionary by an OWNER command, being a string of up to 32 printable characters.

If *owner-name* is delimited (put in single or double quotes) it can include any printable character except the quotation mark. A space character is regarded as a printable character.

specific-level is the specific security level of the user who is identified by user name, when dealing with members owned by the owner who is identified by owner-name. Specific-level is an integer in the range of 0 to 254.

Optionally, up to 255 owner-name entries can be specified following the OWNER keyword in each IDENTITY clause, subject to the logical blocksize of the repository's Data Entries dataset. The specified user can then have access to data belonging to the specified owner. The maximum number of owners is calculated by this algorithm, rounded down to the nearest integer:

(logical_blocksize - 75)/2

For example, if the logical blocksize is 475 bytes, the maximum number of owners is 200. This total is arrived at this way:

(475 - 75)/2 =200

Note:

Ownership of data is established by PROTECT commands.

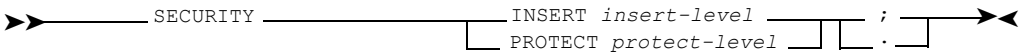
If an owner-name declared in an IDENTITY clause has not been previously recorded in the data dictionary (by means of an OWNER command), the message:

```
owner-name OWNER NOT IN DICTIONARY
```

is output. The owner-name and specific-level are not recorded against the user-name and processing continues with the next owner-name or keyword.

Establishing Insert and Protect Levels in a Dictionary

To establish a security level which must be equalled or exceeded by a user's general security level if updating or PROTECT commands are to be accepted from that user, enter:



where *insert-level* is the security level which must be equalled or exceeded by a user's general security level if any of these updating commands are to be accepted from that user:

ADD	MODIFY
ALTER	REMOVE
BULK ENCODE	RENAME
COPY	REPLACE
ENCODE	CONVERT
INSERT	MERGE

The insert-level is an integer in the range 0 to 254.

protect-level is the security level which must be equalled or exceeded by a user's general security level if PROTECT commands are to be accepted from that user, being an integer in the range 0 to 254.

The insert-level must not be specified more than once in any SECURITY command. It remains in force for comparison with any user's general security level until altered by a later SECURITY command having an INSERT clause.

The protect-level must not be specified more than once in any SECURITY command. It remains in force for comparison with any user's general security level until altered by a later SECURITY command having a PROTECT clause.

Establishing Protect Levels in a Dictionary

Refer to "Establishing Insert and Protect Levels in a Dictionary" on page 195 for details of how to establish protect levels in a dictionary.

Establishing the Default Period of Locks in a Dictionary

To specify the default period of locks for a dictionary, enter:

```
SECURITY RETENTION-PERIOD integer HOURS / DAYS ;
```

where *integer* must not exceed 168 if the keyword HOURS is used and must not exceed 366 if the keyword DAYS is used.

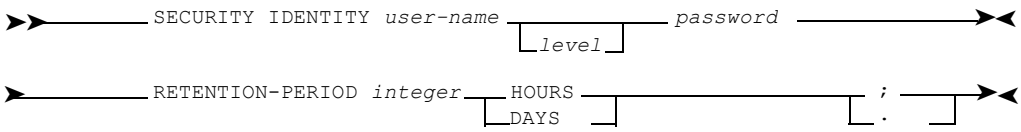
This variant of the SECURITY command is used to establish or alter the length of time after which all locks will expire, providing no different lock periods have been specifically defined for any individual users.

For example, to set the default lock period for the dictionary to twelve hours, enter:

```
SECURITY RETENTION-PERIOD 12 HOURS ;
```

Establishing the Period of Locks for an Individual User

To specify the period of locks for an individual user, enter:



where *integer* must not exceed 168 if the keyword HOURS is used and must not exceed 366 if the keyword DAYS is used.

This variant of the SECURITY command is used to establish or alter the length of time after which a lock will expire, when the LOCK command is entered by the user specified in the IDENTITY clause of this command. If no specific lock period has been specified for a user, the default lock period for the dictionary is applied.

For example, to set the lock period for the user (or department) ACCOUNTS to three hours, where the security level of ACCOUNTS is one and the password is DEPT8, enter:

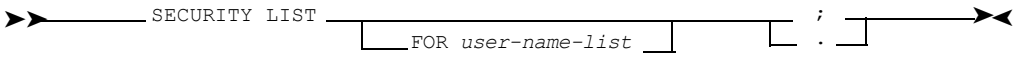
```
SECURITY IDENTITY ACCOUNTS 1 DEPTB RETENTION-PERIOD 3 HOURS ;
```

To delete a user's individual lock period, set the lock period in the RETENTION-PERIOD clause to 0 HOURS.

This variant of the SECURITY command is not available to dictionary Controllers.

Listing Security Information

To list users' security records, enter:



where *user-name-list* is a list of one or more users whose security record you want to display. If two or more user-names are listed, each except the last in the list must be followed by a comma. Entries in the list may additionally be separated by spaces.

If LIST is included in the SECURITY command, a list of users and associated security information is output after all other clauses in the command have been processed. The users are listed in the order in which their records were input to the dictionary. If the LIST keyword is repeated in the SECURITY command, it is actioned only once.

For each user, the password is printed in hexadecimal representation and in character form.

If a user's record is deleted, a deleted-user record is substituted, so as to retain a complete file of names of users who have at any time had access to the dictionary. Deleted users appear in any subsequent SECURITY LIST output this way:

```
PASSWORD IS DELETED FOR USER 'user-name'
```

If the ControlManager Audit and Security facility is installed (selectable unit CMR-DD3), the output from the SECURITY LIST command will additionally include information relating to owners and PROTECT and INSERT levels.

If the ControlManager Workstation Interface—PC and Mainframe Tailoring facility is installed (selectable unit CMR-WS01), you can display users' security records, together with a list of the members each user has locked, by entering:

```
SECURITY LIST FOR user-name-list GIVING LOCKS ;
```

For example, to list the security records of the users (or departments) ACCOUNTS and PERSONNEL, listing the members which each of these users have locked as well, enter:

```
SECURITY LIST FOR ACCOUNTS, PERSONNEL GIVING LOCKS ;
```

Example Output of the Basic SECURITY LIST Command

Figure 8 shows the output received from the basic SECURITY LIST command:

Figure 8 • Output of basic Security List command

```
SECURITY CONTROL INFORMATION
D8D4D7F9                  'LINK'          IS PASSWORD FOR MASTER OPERATOR
D7D4C7D9ADEFCC76          'PMGR'          IS PASSWORD FOR USER 'JOHNSON'
D7D9D6F1                  'LEADER'        IS PASSWORD FOR USER 'ALLEN'
```

If a user's record has been deleted, a deleted-user record is substituted, so as to retain a complete file of names of users who have at any time had access to the dictionary. Deleted users appear in any subsequent SECURITY LIST output thus:

PASSWORD IS DELETED FOR USER *'user-name'*

Example Output of the SECURITY LIST Command with the Audit and Security Facility

Assuming appropriate CREATE, DICTIONARY, AUTHORITY (quoting the Controller's password), and OWNER commands have already been issued, when you enter these commands, ensuring that the LIST keyword is present:

```
SECURITY INSERT 100 PROTECT 110
IDENTITY 'JOHNSON' 200 'PMGR
OWNER 'PAYROLL' 200, 'ACCOUNTS' 180
IDENTITY 'ALLEN' 100 'PRO1'
IDENTITY 'SMITH.A' 80 'PRO2'

SECURITY IDENTITY 'HAMILTON' 110 'SPRO'
OWNER 'PAYROLL' 150 LIST;
```

the resulting output would be as shown in Figure 9 on page 199:

Figure 9 • Output of SECURITY LIST Command with Audit and Security Facility

```
SECURITY CONTROL INFORMATION
PROTECTION LEVEL 100                INSERTION LEVEL 110
 D7D4C7D9AOEFCC76 'PMGR     '       IS PASSWORD FOR USER JOHNSON
      LEVEL      OWNER
      200        *****
      200        'PAYROLL'
      180        'ACCOUNTS'

D7D9D6F1         'PRO1'             IS PASSWORD FOR USER 'ALLEN'
      LEVEL      OWNER
      100        *****

D7D9D6F2         'PRO2'             IS PASSWORD FOR USER 'SMITH,A'
      LEVEL      OWNER
      80         *****

E2D7D9D6         'SPRO'             IS PASSWORD FOR USER 'HAMILTON'
      LEVEL      OWNER
      110        *****
      150        'PAYROLL'
```

After the heading, the protect-level and the insert-level are printed, followed by the list of users. For each user the password is printed in hexadecimal representation and in character form, the security-levels are then listed; first, the user's general security-level (denoted by *****), then the user's specific security-level in relation to each owner to whose data definitions access has been authorized by the IDENTITY clause of the SECURITY command.

Note:

The password of the user Johnson includes non-printable characters.

Example Output of the SECURITY LIST Command with the Workstation Interface Facility

Assuming appropriate CREATE, DICTIONARY, AUTHORITY (quoting the Controller's password), and OWNER commands have already been issued, when you enter this command:

```
SECURITY LIST FOR PR05. PR07 GIVING LOCKS ;
```

output similar to Figure 10 on page 200 displays:

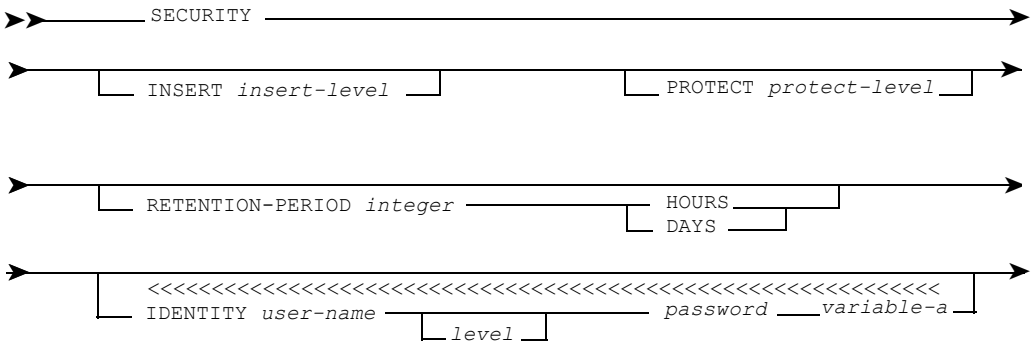
Figure 10 • Output of SECURITY LIST Command with Workstation Interface Facility

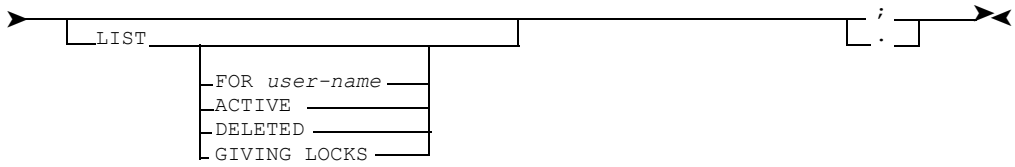
```
SECURITY CONTROL INFORMATION FOR SELECTION GIVING LOCKS
PROTECTION LEVEL 100           INSERTION LEVEL 100
                          DEFAULT LOCK-RETENTION PERIOO IS 12 HOURS
C7D4C7D9ADEFCC76  '.PR05'      IS PASSWORD FOR USER 'JACKSON.W'
                               USER'S LOCK-RETENTION PERIOD IS 48 HOURS


LIST OF LOCKED MEMBERS
EXPIRY DATE     EXPIRY TIME     MEMBER NAME     MEMBER TYPE     STATUS
01 MAY 1999     14.04.12        MEM3            FILE            LIVE
30 APR 1999     11.33.20    EXPIRED ACCT-NO     ITEM            LIVE
30 APR 1999     10.58.19    EXPIRED PERS-NO     ITEM            DEV1
28 APR 1999     08.45.30    EXPIRED NEW-LIST    GROUP           LIVE
LEVEL      OWNER
200        *****
200        'PAYROLL'
180        'ACCOUNTS'
C7D9D6F1       'PRO7'    IS PASSWORD FOR USER 'DAVIES.M'
                         USER'S LOCK-RETENTION PERIOD IS 6 HOURS

LIST OF LOCKED MEMBERS
EXPIRY DATE     EXPIRY TIME     MEMBER NAME     MEMBER TYPE     STATUS
02 MAY 1999     18.24.42        MSP-DIST        ITEM            DEV1
02 MAY 1999     18.00.32        DEPT3           GROUP           LIVE
29 APR 1999     14.35.10        EXPIRED MEMO36  FILE            LIVE
LEVEL      OWNER
100        *****
END OF SECURITY LIST
```

The Maximum Number of User Security Records Allowed

The maximum number of users whose security information can be recorded in the dictionary is 65535 (in addition to the Controller). This maximum includes the records of any deleted users.

SECURITY Syntax





where:

insert-level is an integer in the range 0 to 254.

protect-level is an integer in the range 0 to 254.

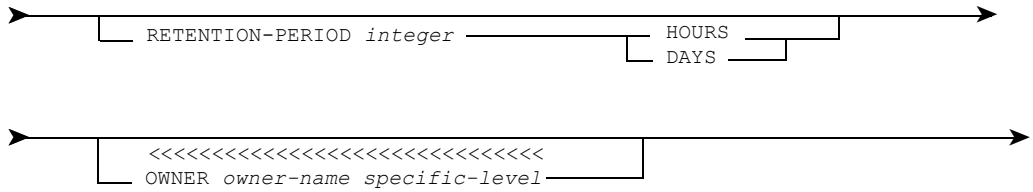
integer is an integer in the range 1 to 168 if the keyword HOURS is used, or 1 to 366 if the keyword DAYS is used.

user-name is a string of up to 32 printable characters.

level is an integer in the range 0 to 254, with a default of 1 if omitted.

password is a character string of up to 8 printable or non-printable characters.

variable-a is:



where:

integer is as defined above.

owner-name is a string of up to 32 printable characters.

specific-level is an integer in the range 0 to 254.

Note:

In order for the SECURITY command to be accepted, the command identifier must be followed by at least one of the options specified above.

SET ENCODE-EXIT

This command enables additional checking of dictionary members' sources whenever the encode process takes place, by calling user tailorable executive routines.

To enable the user exit which calls the executive routine MPCMENCUX when a dictionary member is encoded, issue the command:

```
SET ENCODE-EXIT ON ;
```

The default setting is OFF.

If the ENCODE-EXIT is on, then, if the standard syntax and entity relationship checks complete successfully, the executive routine MPCMENCUX is called. Within this executive routine, the user can DACCESS the newly encoded member and/or do other interrogates to further validate the member, but may not perform dictionary updates. MPCMENCUX can be further tailored to output any relevant diagnostic messages and to carry out further updates within the exit by use of a second exit routine.

Further processing depends on the return code. MPCMENCUX return codes translate this way:

Return code	Translation
0	Accept the encode and finish normally
4	Reject the encode and finish normally
8	Accept the encode, then call a second cexec to do further processing
12	Reject the encode, then call a second cexec to do further processing
16	Same as for return code 12, but the output from the first encode and any user exit output is hidden

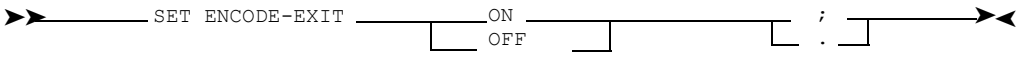
The name of the second exit routine should be set in the global variable MPCM_ENCODE EXIT_2 during the execution of MPCMENCUX. Any information to be saved by the user between the two execs must be stored in global variables.

The printed output from any command issued by the second executive routine is under your control; NOPRINT should be used to suppress output if required. The second executive routine is allowed to do dictionary updates but these updates will NOT call any ENCODE-EXIT routines. You cannot change or create dictionaries within the executive routine or instigate a LUW.

Exclusive control of the dictionary is maintained throughout both exits until the originating command is terminated; however updates issued within the second executive routine are committed/rolled back and logged as individual commands unless the original command is within a LUW.

No return code is expected from the second executive routine.

SET ENCODE-EXIT Syntax



SHOW UDS

The SHOW UDS command displays all or part of the contents of a UDS table.

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for documentation of the SHOW UDS command. The TABLE keyword is restricted to the repository Controller.

STATUS DEFAULT

The STATUS DEFAULT command specifies the default status for the dictionary, enter:

```
STATUS DEFAULT  status-name ;
```

where *status-name* is the name of a named status that already exists in the dictionary.

Alternatively, if the status you last froze is the new default status, you can enter:

```
STATUS DEFAULT LAST-FROZEN-STATUS ;
```

A STATUS/DEFAULT command specifies the default status for the dictionary. When a DICTIONARY command is accepted, the default status is automatically invoked as the current status; that is, the status in which Manager Products operate. Thereafter, users may specify a new current status by successfully executing a STATUS *status-name* command.

If the status name is specified in a STATUS DEFAULT command, the specified status becomes the default status for the dictionary, until:

- Another STATUS DEFAULT command is received for that dictionary
- A STATUS MERGE command affecting the default status is successfully processed (the resulting merged status becomes the new default status)
- A STATUS REMOVE command affecting the default status is successfully processed. The dictionary then behaves as if no default status has been specified

The status name in the STATUS DEFAULT command must already have been recorded in the dictionary by a STATUS NAME command. If the specified default status is renamed, it will remain the default status under its new name.

If LAST-FROZEN-STATUS is specified in a STATUS DEFAULT command, the default status is whichever status is the most recently frozen; until a further STATUS DEFAULT command is received for the dictionary. If no frozen status exists in the dictionary, the first named status is taken as the default status until a status is frozen or a further STATUS DEFAULT command is received for the dictionary.

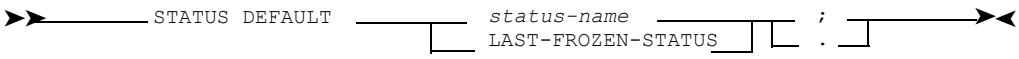
If no STATUS DEFAULT command has been received for a dictionary, the default status is the earliest frozen status in the dictionary, that is, the root status; or if no frozen statuses exist in the dictionary, then the first named status.

If no statuses have yet been named in the dictionary, Manager Products operate in a default unnamed status. If statuses are subsequently named, the first status name accepted is applied to that previously unnamed default status.

STATUS DEFAULT is treated as an updating command by the automatic error recovery system.

Following successful execution of a STATUS DEFAULT command dictionary users must specify, or respecify, the current status, in order to continue using the dictionary.

STATUS DEFAULT Syntax



where *status-name* is a character string of not more than 32 characters.

STATUS FREEZE

The STATUS FREEZE command designates the position of a status in the status hierarchy.

The command:

```
STATUS FREEZE status-name ;
```

causes the specified status to become a read-only base status. If the dictionary already contains base statuses then the specified status will become the direct dependent of the most recently frozen base status and the direct base of all non-frozen statuses in the dictionary (if it has any). If the dictionary does not already contain a base status; that is, if no other status has been frozen, the specified status will be the root status.

Any remaining non-frozen statuses in the dictionary are directly based upon the newly frozen status. To achieve this, ControlManager has to adjust the used-by and references pointers of any members in those statuses, by re-encoding all encoded members in any non-frozen status where such adjustment is necessary.

During re-encoding, messages are output if the re-encoding of individual members fails. Alias validation rules specified using the CONTROL NEW-ALIASES command are applied. Warning messages only are issued for members which contravene rules on alias length, and which fail to include mandatory alias types, but these will not cause the re-encode to fail. However, checking for alias names which are duplicates of member names or aliases will be applied across visible statuses; if errors are found, error messages are issued and the re-encoding fails. No messages are output in respect of members successfully re-encoded.

If you are using the Basic Status facility then all base statuses are read-only statuses. If you are using the Advanced Status facility then frozen statuses are read-only but may be changed to update statuses (and back again) at any time using the STATUS PERMIT command or, for individual and/or groups of users, the Status Window.

Statuses may be frozen in any order, but as indicated above, the order in which you freeze them determines the structure of the status hierarchy.

If a status is successfully frozen, the message:

```
STATUS status-name FROZEN AT hh.mm.ss ON dd.mmm.yy
```

is output. The time and the date it was frozen are stored in the dictionary and are included in output from STATUS LIST commands as the date and time upon which the status became a read-only status. If the status is subsequently changed to an update status then this date and time record is overwritten.

If the status name in a STATUS FREEZE command is already a base status, the message:

```
STATUS status-name ALREADY FROZEN
```

is output, and the command has no other effect.

STATUS FREEZE and STATUS UNFREEZE commands both involve adjusting of used-by and reference pointers and may involve much processing time. For this reason repeated sequences of FREEZE/UNFREEZE/FREEZE operations are not recommended.

If you are using the Basic Status facility (which does not allow updatable base statuses) you should be particularly careful to ensure that all members are correct in the relevant status before issuing a STATUS FREEZE command, so that there is no need to issue a subsequent STATUS UNFREEZE command in order to make corrections.

Note:

The contents of a non-base status can be effectively frozen by merging it into its direct base using STATUS MERGE.

If the command takes a long time to process you should UNLOAD the dictionary immediately afterward in order to avoid the need to re-execute the command during ROLL-FORWARD. This also applies to STATUS UNFREEZE and STATUS MERGE commands.

The dictionary cannot be used during execution of a STATUS FREEZE command and dictionary users must specify, or respecify, the current status, in order to continue using the dictionary following its successful completion.

To find out which statuses will be re-encoded as the result of your freezing a particular status, this procedure is recommended:

- Access the status to be frozen and enter:

```
KEEP LIST CURRENT DEFINITIONS ;
KEEP LIST CURRENT USAGE-TABLES ;
```

This builds a KEPT-DATA list out of all members present in the status except those which are present as source records only.

- Access each remaining non-frozen status in turn and enter:

```
KEEP IN kept-list-name LIST CURRENT DEFINITIONS KEPT;
ALSO IN kept-list-name LIST CURRENT USAGE-TABLES KEPT;
```

where *kept-list-name* is the name of the status in which the commands are entered. The resulting named KEPT-DATA lists will contain the members which exist in each non-frozen status as encoded records or usage-tables that also exist in the unnamed KEPT-DATA list and, therefore, in the status you are going to freeze. Then enter:

```
QUERY KEPT ALL;
```

and the output will show zero against statuses that do not overlap and will not, therefore, be re-encoded as a result of the proposed freeze.

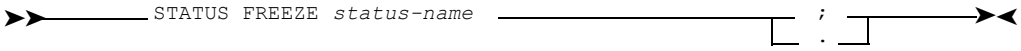
Note:

An update to a single member could invalidate the results of the above procedure. Consequently, it is advisable to disable the dictionary (using the DISABLE command) while it is being carried out and up until the required status is frozen.

In order to use the above procedure, ASG-DataManager, which provides the KEPT-DATA capabilities, must be installed.

STATUS FREEZE is a restartable command.

STATUS FREEZE Syntax



where *status-name* is a character string of not more than 32 characters.

STATUS LIST

The STATUS LIST command is used to obtain a detailed list and/or diagram representing the statuses in the dictionary.

The STATUS LIST command available to dictionary Controllers is an expanded version of that available to all users. The additional keywords provided (DICTIONARY, BASE-OF, and DEPENDENT-OF) are documented here. For the description of the STATUS LIST command available to all users, and illustrations of its output, please refer to *ASG-Manager Products Basic Status* or *ASG-Manager Products Advanced Status*.

If the Advanced Status facility is installed, the Systems Administrator may limit the range of statuses that individuals and/or groups of users can list. This is done by placing the appropriate variant of the STATUS WINDOW MAXIMUM command into Global Profiles, Logon Profiles, and/or Corporate Executive Routines.

To include in the output list the number of unnamed statuses and the number of named statuses in the dictionary, regardless of any limitations set through the STATUS WINDOW MAXIMUM/MINIMUM command, enter:

```
STATUS LIST DICTIONARY ;
```

To include in the list output, for each listed status, a sub-list of the statuses of which it is a base, enter:

```
STATUS LIST BASE-OF ;
```

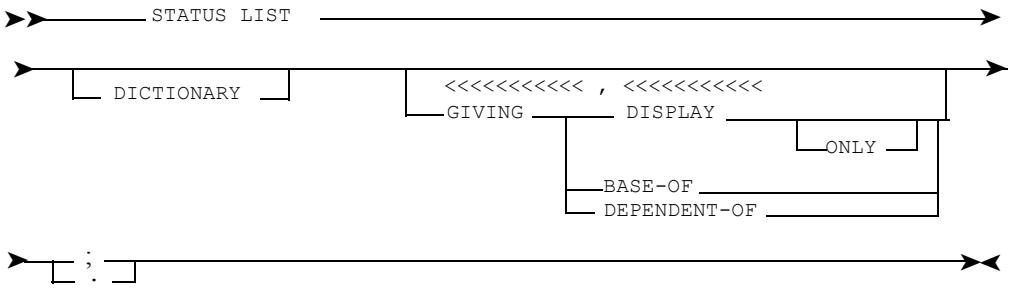
The first direct dependent status appears at the top of the sub-list followed by its dependent and so on in hierarchical sequence (that is, top-down, left-to-right).

To include in the list output, for each listed status, a sub-list of the statuses upon which it is dependent, enter:

```
STATUS LIST DEPENDENT-OF ;
```

The root status appears at the top of the sub-list and the direct base status at the bottom.

STATUS LIST Syntax



STATUS MERGE

The STATUS MERGE command combines two statuses, typically to merge a status used for development work into a production status.

To combine two statuses enter:

```
STATUS MERGE status-name INTO status-name-b ;
```

The statuses are merged under the name *status-name-b*, and *status-name* is removed from the dictionary. To merge the statuses under a different name, enter:

```
STATUS MERGE status-name INTO status-name-b
RENAMED-AS status-name-c ;
```

where *status-name-c* is subject to the rules governing status names. It must not already exist in the dictionary—unless it is the name of one of the two statuses being merged. The statuses are merged under the name *status-name-c*, and both *status-name* and *status-name-b* are removed from the dictionary.

Statuses which fall into one of these categories can be merged:

- Two statuses which have a direct relationship; that is, each is the direct base or the direct dependent of the other
- Sibling statuses; that is, dependent statuses with the same direct base status, provided that they are not diverging

See "Merging Sibling Statuses" on page 59 for details of merging sibling statuses.

Where the statuses being merged are not siblings and there are differences in the definitions of a member recorded in each, then:

- If both are base statuses, the definition in the most recently frozen of the two is taken into the merged status
- If a non-base status and the most recently frozen base status are being merged, the definition in the non-base status is taken into the merged status

If the statuses named in a STATUS MERGE command are both base statuses but are not adjacent in the status hierarchy, the message:

```
STATUSES FOR MERGING NOT ADJACENT
```

is output, and the command is rejected.

If one of the statuses named in a STATUS MERGE command is a non-base status, and the other is a base status but not its direct base (that is, it is not the most recently frozen base status) the message:

STATUS *status-name* NOT MOST RECENTLY FROZEN

is output, and the command is rejected.

If two statuses are successfully merged, the message:

STATUSES *status-name* AND *status-name-b* MERGED UNDER NAME *status-name-b*

(or UNDER NAME *status-name-c*) is output.

The date and time recorded for the naming of the merged status is the earlier of the dates of naming of *status-name* and *status-name-c*.

ControlManager keeps a date and time record for each status designated as a read-only status. A successful STATUS MERGE command affects these read-only date and time records this way:

- If *status-name* is a read-only status and *status-name-b* is an update status, the merged status will be an update status and the read-only date and time record for *status-name* is erased
- If *status-name* is an update status and *status-name-b* a read-only status, the merged status will be a read-only status, and the read-only date and time record will reflect the date on which the STATUS MERGE command was run
- If the statuses being merged are both read-only statuses, the later of the date and time records recorded for each is taken as the read-only date and time record for the merged status; the read-only date and time records for the statuses being merged are erased.

If either *status-name* or *status-name-c* was the default status, the status resulting from a successful STATUS MERGE command becomes the new default status.

STATUS MERGE is not included among the updating commands from which the automatic error recovery system can recover. Before issuing a STATUS MERGE command, therefore, you should take a backup copy of the dictionary.

If a STATUS MERGE command takes a long time to process, you should UNLOAD the dictionary immediately afterward in order to avoid the need to reexecute the command during ROLL-FORWARD. This also applies to STATUS FREEZE and STATUS UNFREEZE commands.

The dictionary cannot be used during execution of a STATUS MERGE command and dictionary users must specify, or respecify, the current status in order to continue using the dictionary following its successful completion.

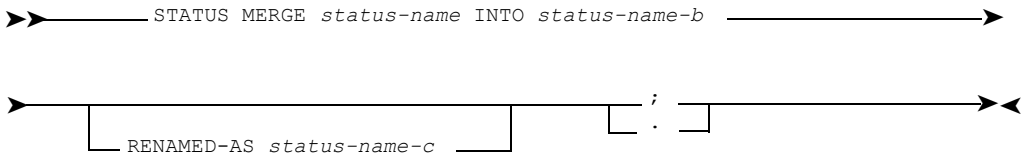
If you are using Basic Status, all base statuses are read-only statuses. You can use this procedure to effect an update to the most recently frozen of those base statuses:

- Name a new non-base status (this will be designated an update status by default)
- Working in the non-base status, complete the required updates to the required members
- Merge the non-base status into the most recently frozen base status

This procedure is not relevant if you are using Advanced Status since it allows members in base statuses to be updated (if that status is designated as an update status via STATUS PERMIT and a STATUS WINDOW MINIMUM/MAXIMUM command).

Since a STATUS MERGE command removes and may add status names from and to the dictionary, you should consider the effect that this may have upon Profiles and Executive Routines which include STATUS status-name commands and, if you are using Advanced Status, STATUS WINDOW commands.

STATUS MERGE Syntax



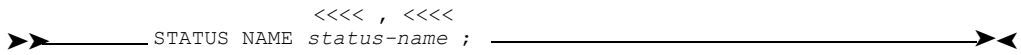
where:

status-name, *status-name-b*, and *status-name-c* are character strings of not more than 32 characters.

STATUS NAME

A STATUS NAME command assigns names to statuses. If the number of status names in the command exceeds the number of unnamed statuses in the dictionary, the excess names are rejected. Unnamed statuses are created when the dictionary itself is created.

To assign names to unnamed statuses, enter:



where *status-name* is subject to the rules governing status names. These are stated in *ASG-ControlManager User's Guide*.

Since a STATUS NAME command adds status names to the dictionary, you should consider the effect that this may have upon Profiles and Executive Routines which include STATUS *status-name* commands and, if you are using Advanced Status, STATUS WINDOW commands.

If a status is successfully named, the message:

```
STATUS status-name NAMED AT hh.m.ss ON dd.mmm.yy
```

is output. The date and time of naming are stored in the dictionary.

If a status name is rejected because there are no unnamed statuses remaining in the dictionary, the message:

```
status-name REJECTED — ALL STATUSES ARE ALREADY NAMED
```

is output.

If a status name in the STATUS NAME command already exists in the dictionary, the message:

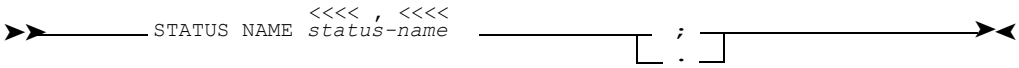
```
status-name REJECTED - THIS STATUS ALREADY EXISTS
```

is output.

STATUS NAME is treated as an updating command by the automatic error recovery system.

You are advised not to name a status until it is needed, because unused named statuses may cause unnecessary processing overheads.

STATUS NAME Syntax



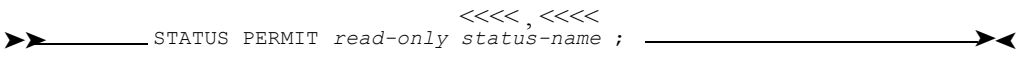
where *status-name* is a character string of not more than 32 characters.

STATUS PERMIT

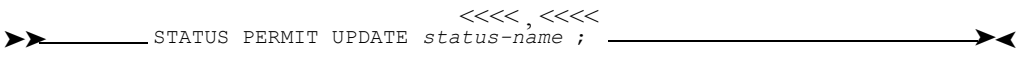
The STATUS PERMIT command enables you to change the read-only or update condition of a status. The command may be issued while the dictionary is in use and its effect is immediate.

Base (frozen) statuses are read-only and non-base (unfrozen) statuses are update by default. If the Basic Status facility is installed you can use the STATUS PERMIT command to designate non-base (unfrozen) statuses as read-only or update statuses. If the Advanced Status facility is installed you can use the STATUS PERMIT command to designate any status as read-only or update.

To designate statuses as read-only, enter:



To designate statuses as update, enter:



The command will be accepted or rejected independently for each status name specified.

READ-ONLY and UPDATE clauses can both be present in the same command. If the same status is specified several times in a command only the first occurrence will be actioned and an error message will be output for each subsequent occurrence.

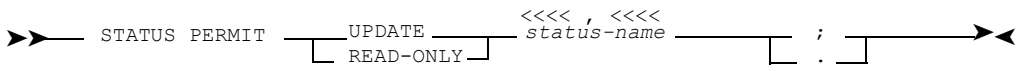
Once the command is executed all users accessing the dictionary will have their status details updated to reflect the change.

If you are using the Advanced Status facility, you can establish the read-only/update condition of a status for individuals and/or groups of users via the Status Window command.

Examples

```
STATUS PERMIT UPDATE HITS, LIVE ;
STATUS PERMIT UPDATE LIVE READ-ONLY DEV1 ;
```

STATUS PERMIT Syntax



where *status-name* is a character string of not more than 32 characters.

STATUS PURGE-DATA-ENTRIES

The STATUS PURGE-DATA-ENTRIES command removes data entries dataset records from a status, enter:

```
STATUS PURGE-DATA-ENTRIES status-name ;
```

The STATUS PURGE-DATA-ENTRIES command may be applied to non-base statuses only. It deletes from the data entries dataset the records of all members in the specified status. The status becomes a source-only status (until such time as members are again encoded in that status).

As the command completely clears the data entries dataset in the specified status, it should be used with great caution.

As the data entries information is removed, the record of the date each member was last encoded in the status, and by which user, is also removed.

STATUS PURGE-DATA-ENTRIES is not included among the updating commands from which the automatic error recovery system can recover. You should, therefore, back up your dictionary before issuing a STATUS PURGE-DATA-ENTRIES command.

The removal of the records from the data entries dataset makes it easier to rename members, to remove members, and/or to reorganize the hierarchical relationships between them, where such tasks would otherwise be rendered difficult by the relationships set up when the members were encoded. However, this does involve having to encode all the members subsequently.

The removal of the records from the data-entries dataset can significantly reduce the time it takes to freeze, merge, and unfreeze statuses. Although the total processing time taken, including subsequently re-encoding the required members, may be greater, the length of time during which the dictionary is unavailable for use is reduced.

This procedure may be used as a preliminary to freezing, merging, or unfreezing statuses:

1. Find out which statuses have diverging definitions for the same members when compiled to the status to be frozen/merged/unfrozen. See Chapter 8, "Status Facilities for Controllers," on page 53 for how to do this.
2. Apply a STATUS PURGE-DATA-ENTRIES command to each of those statuses.
3. Apply the STATUS FREEZE/MERGE/UNFREEZE command as required.
4. Re-encode all members in the purged statuses.

The re-encoding process could be carried out as a batch job the following night or in stages during the day with the dictionary available to other users for some of the time.

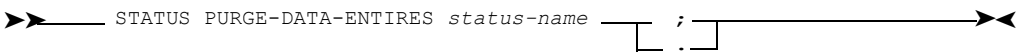
One other advantage of using the above procedure is that the re-encoding process (using BULK ENCODE or PERFORM ENCODE selection...' commands), if terminated abnormally, can be restarted from the point at which it terminated. To do this, enter:

`BULK ENCODE UNVERIFIED` or `BULK ENCODE FROM` *`last-name`*

where *`last-name`* is the name of the member at which the re-encoding process terminated.

Processing of STATUS MERGE commands, if terminated abnormally, would need to be restarted from the beginning after the dictionary has been RELOADed.

STATUS PURGE-DATA-ENTRIES Syntax



where *`status-name`* is a character string of not more than 32 characters.

STATUS REMOVE

The STATUS REMOVE command removes a root status or a non-base status from the dictionary, enter:

```
STATUS REMOVE status-name ;
```

A STATUS REMOVE command is accepted only if the specified status is a root status or a non-base status, and the effect of the command differs accordingly.

A root status (upon which other statuses are based) may only be removed if its direct dependent is a base status; that is, the root status must not be the only base status in the dictionary. When a root status is successfully removed, the definitions it contains are merged into the direct dependent status and the root status name is removed from the dictionary. (The recorded times and dates of naming and freezing of the dependent status are left unchanged.) Where there are differences in the data definitions of a member in the two statuses, the definition in the dependent status is taken into the merged status and the definition from the root status is discarded.

If the specified root status is the only base status in the dictionary, the message:

```
STATUS status-name IS THE ONLY BASE STATUS
```

is output, and the command is rejected.

If the specified status is a base status but is not also a root (earliest frozen) status, the message:

```
STATUS status-name IS NOT A ROOT STATUS
```

is output, and the command is rejected.

If the specified status is a non-base status, that status name and all members' records in that status are removed from the dictionary. A root status which is also a non-base status is treated as a non-base status by the STATUS REMOVE command.

Whether the specified status is a root status or a non-base status, a successfully executed STATUS REMOVE command creates an additional unnamed status in the dictionary available to the dictionary Controller for naming.

When a non-base status is successfully removed, a series of messages are output listing any members that no longer exist in the dictionary because they existed only in the removed status.

If a status is successfully removed, the message:

```
STATUS status-name HAS BEEN REMOVED
```

is output; unless the removed status was the default status, in which case the messages:

```
DEFAULT STATUS status-name HAS BEEN REMOVED, NEW DEFAULT STATUS IS
status-name-2
```

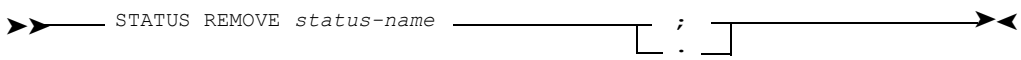
are output, where *status-name-2* is the name of the most recently frozen status or, if no statuses have been frozen, the first named status.

STATUS REMOVE is not included among the updating commands from which the automatic error recovery system can recover. You are therefore advised to back up the dictionary before issuing the command.

Following execution of a STATUS REMOVE command, dictionary users must specify or respecify the current status in order to continue using the dictionary.

Since a STATUS REMOVE command removes status names from the dictionary, you should consider the effect that this may have upon Profiles and Executive Routines which include STATUS status-name commands and, if you are using Advanced Status, STATUS WINDOW commands.

STATUS REMOVE Syntax



where *status-name* is a character string of not more than 32 characters.

STATUS RENAME

A STATUS RENAME command changes the name of the status from *status-name* to *status-name-b*. Any status can be renamed.

To change the names of statuses enter:

where *status-name-b* is subject to the rules governing status names. These are stated in *ASG-ControlManager User's Guide*.

If a status is successfully renamed, the message:

```
STATUS status-name RENAMED AS status-name-b
```

is output. The date and time of naming and, if applicable, of freezing of *status-name* are carried over to *status-name-b*.

If *status-name-b* already exists in the dictionary, the command is rejected and the message:

```
STATUS status-name NOT RENAMED— STATUS status-name-b ALREADY EXISTS
```

is output.

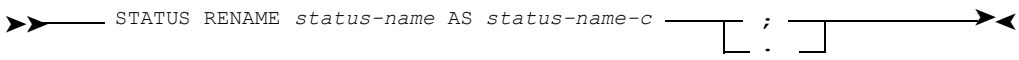
If the default status is renamed, it remains the default status under its new name.

STATUS RENAME is treated as an updating command by the automatic error recovery system.

Following successful execution of a STATUS REMOVE command, dictionary users must specify or respecify the current status in order to continue using the dictionary.

Since a STATUS RENAME command both adds and removes status names to/from the dictionary, you should consider the effect that this may have upon Profiles and Executive Routines which include STATUS *status-name* commands and, if you are using Advanced Status, STATUS WINDOW commands.

STATUS RENAME Syntax



where:

status-name is a character string of not more than 32 characters.

status-name-c is a character string of not more than 32 characters.

STATUS status-name

To find out which is the current status, enter:

```
STATUS ;
```

The current status is the status in which you are working at any one point in time. To change the current status, enter:

```
STATUS status-name ;
```

When STATUS only is specified, a message is output telling you the current status' name, the date and time it was named, and if it is a read-only or an update status. For further details of status date and time records refer to documentation of output from the STATUS LIST command. (See *ASG-Manager Products Basic Status* or *ASG-Manager Products Advanced Status*.)

When a STATUS *status-name* command is successfully processed, a message is output telling you that the specified status has been recognized and whether it is an update status or a read-only status.

As dictionary Controller you can specify a default status for the dictionary, using the STATUS DEFAULT command. The default status is automatically assumed as the current status by the DICTIONARY command. Consequently, users accessing the dictionary need not specify the current status using the STATUS *status-name* command, unless they want to move to another status. A message is displayed when the dictionary is accessed telling the user that the default status has been assumed and giving its name.

The current status is unspecified in these circumstances:

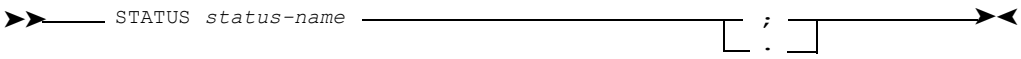
- An invalid status name is specified in a STATUS *status-name* command
- As the result of the execution of any of these commands:
  - STATUS DEFAULT
  - STATUS FREEZE
  - STATUS MERGE
  - STATUS PERMIT
  - STATUS REMOVE STATUS RENAME
  - STATUS UNFREEZE

When the current status is unspecified, ControlManager will not accept any dictionary management commands from general users or from the dictionary Controller. So, until the current status is specified (successfully) the dictionary is effectively unusable. Consequently, following execution of the commands listed above, each dictionary user including the Controller must respecify the current status in order to carry on using the dictionary.

Note:

This does not necessarily mean that users must enter a valid STATUS *status-name* command. Execution of a Global Profile, Logon Profile, User Defined Profile, or Executive Routine containing a valid STATUS *status-name* command, or a valid DICTIONARY and related AUTHORITY command, will have the same effect.

STATUS Syntax



where *status-name* is a character string of not more than 32 characters.

STATUS UNFREEZE

Whereas a STATUS FREEZE command causes the specified status to become a base status, the STATUS UNFREEZE command changes the specified base status into a non-base status. It changes the position of the other non-base statuses in the dictionary so that they are no longer based upon it, but instead upon the most recently frozen of the remaining frozen statuses. See "Building and Maintaining Status Structures" on page 56 for a general discussion about maintaining the status hierarchy.

To change a base status into a non-base status and thus change the position of a status in the status hierarchy, enter:

```
STATUS UNFREEZE status-name ;
```

The command may only be applied to the most recently frozen status. If it is not so applied this message is output:

```
STATUS status-name NOT LATEST FROZEN
```

When a status is successfully unfrozen, the next most recently frozen status (if there is one) becomes the status on which all non-frozen statuses are based. To achieve this, ControlManager has to adjust the used-by and reference pointers of members in non-frozen statuses (excluding status-name) by re-encoding all encoded members in any non-frozen status in which such adjustment is necessary.

During re-encoding, messages are output if the re-encoding of individual members fails. Alias validation rules specified using the CONTROL NEW-ALIASES command are applied. Warning messages only are issued for members which contravene rules on alias length, and which fail to include mandatory alias types, but these will not cause the re-encode to fail. However, checking for alias names which are duplicates of member names or aliases will be applied across visible statuses; if errors are found, error messages are issued and the re-encoding fails. No messages are output in respect of members successfully re-encoded.

STATUS UNFREEZE and STATUS FREEZE commands both involve processing of used-by and reference pointers (see above). This may involve much processing time. For this reason repeated FREEZE/UNFREEZE/FREEZE operations are not recommended.

All unfrozen statuses are update statuses by default but may be changed to read-only (and back again) at any time with a STATUS PERMIT command.

If a status is successfully unfrozen, this message is output:

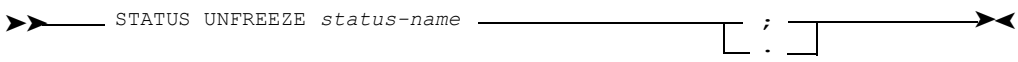
```
STATUS status-name UNFROZEN
```

If the command takes a long time to process you should UNLOAD the dictionary immediately afterward in order to avoid the need to re-execute the command during ROLL-FORWARD. This remark also applies to STATUS FREEZE and STATUS MERGE commands.

The dictionary cannot be used during execution of a STATUS UNFREEZE command and dictionary users must specify, or respecify, the current status in order to continue using the dictionary following its successful completion.

STATUS UNFREEZE is a restartable command. (See "Restartable STATUS FREEZE and STATUS UNFREEZE" on page 60.)

STATUS FREEZE Syntax



where *status-name* is a character string of not more than 32 characters.

STATUS WINDOW MAXIMUM/MINIMUM

The STATUS WINDOW command is used to control the range of statuses processed when certain related activities are undertaken by the Controller and/or individual dictionary users or groups of dictionary users.

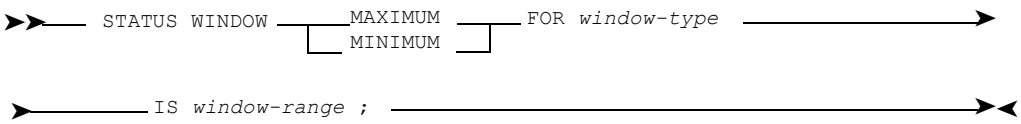
This variant of the STATUS WINDOW command is the only variant documented specifically for dictionary Controllers.

For details of the command variants available to all users, please refer to *ASG-Manager Products Advanced Status*.

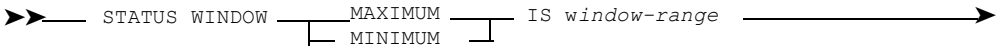
Controller's Use of the Command

This restricted variant of the STATUS WINDOW command enables you to control the range of statuses processed when certain status-related activities are undertaken by yourself and/or individual/groups of dictionary users. For example, it can be used to control the update condition of a status for individual users. Several additional Window Types are available with this form of the command that are not available otherwise.

The format of this variant of the command is:



Alternatively, to specify one maximum or minimum range for the whole Status Window, enter:



where *window-type* and *window-range* are the same as those available to all users with these additional window-type keywords when MAXIMUM is specified:

- READ-ONLY
- UPDATE
- STATUS-LIST

and the additional window-range keyword:

- NONE

(which may be specified only when MAXIMUM or MINIMUM is also specified) and the exception that the window-range:

```
LATEST-VISIBLE-OCCURRENCE
```

may not be specified when MAXIMUM or MINIMUM is specified.

Examples

```
STATUS WINDOW MAXIMUM FOR DIVERGING IS ONLY STATUSES
                          DEV1, DEV2, PROD ;
STATUS WINDOW MAXIMUM FOR SELECTION IS ONLY CURRENT
                          AND STATUS DEV2
FOR OUTPUT IS ROOT ;
```

Tailoring a User's Status Environment

The use of MAXIMUM and MINIMUM keywords in the STATUS WINDOW WINDOW command is restricted to use by dictionary Controllers when entered through the Command Area, a User Defined Profile or a User Executive Routine. When they are so used they apply only to the Controller's own Status Window (each dictionary user has a Status Window).

When placed within Logon Profiles, Global Profiles, and Corporate Executive Routines they can be used to control the range of statuses included in a user's Status Window and thereby the range of statuses processed by that user for a number of status-related activities.

When MAXIMUM/MINIMUM is specified, the Status Window controls additional activities that it does not control when they are not specified; that is, when a user adjusts the Status Window.

And, when MAXIMUM is used, several additional Window Types are available, each of which controls the range of statuses processed for a different status-related activity.

The range of statuses that a user can include in the Status Window for Selection, Output, and Diverging may be controlled and thus the range of statuses that are processed by these activities:

- ALL-STATUSES selection
- LIST HISTORY command
- status-related-selection DIVERGING.

When a MAXIMUM Window Range is specified for Selection, the Status Window also controls the range of statuses processed for this activity:

- IF clause in time-and-user-related-selection; for example, IF VERIFIED IN status-name.

A status must be included in the maximum Status Window range in order that the specified selection is applied to it. The maximum range of a Status Window may only be adjusted by a dictionary Controller. It is not affected when a user adjusts his/her Status Window; that is, when MAXIMUM is not specified.

When a MAXIMUM Window Range is specified for Output, the Status Window also controls the range of statuses available for this activity:

- Using the FROM (*status-name*) option in the COPY command.

The specified status must be included in the Status Window in order for the COPY command to be accepted. This activity is not affected when a user adjusts the Status Window; that is, when MAXIMUM is not specified

Additionally the MAXIMUM keyword enables you to adjust a user's Status Window for Update, Read-only, and Status-list (users cannot adjust these Window Types themselves) and thereby to control these things:

- The statuses to which a user has update access
- The statuses to which a user has read-only access
- The output from STATUS LIST commands.

See "Activities Controlled by Maximum and Minimum Window Range" on page 63 for a tabulation showing each Window Type and the activities it controls.

A user's attempt to define a Window Range (using the STATUS WINDOW command) which exceeds the current maximum or minimum Window Range does not result in the rejection of the command. The command is accepted but the maximum or minimum Window Range is applied; the range requested by the user is relative to the maximum or minimum range.

So, a status is included in a user's Status Window if it is within the maximum Window Range, as specified through a STATUS WINDOW MAXIMUM command and it is included in either or both:

- The current Status Window Range as specified by that user through a STATUS WINDOW command
- The MINIMUM Window Range as specified through a STATUS WINDOW MINIMUM command

(The Window Range requested by a user will not exclude, from the Status Window, statuses included within the MINIMUM Window Range.)

For example, a user may specify DICTIONARY in a STATUS WINDOW command; if a STATUS WINDOW MAXIMUM command specifying CURRENT has been executed in the current session, then that is the Window Range effectively defined by DICTIONARY.

As supplied by ASG, the Status Window has a default maximum and minimum Window Range of DICTIONARY for all Window Types.

Since Profiles and Corporate Executive Routines are controlled by the Systems Administrator, you (that is, the dictionary Controller) must contact the Systems Administrator to arrange the profiling/executives you require for the users of your dictionary.

READ-ONLY and UPDATE Options

A status is designated as read-only or update by the STATUS FREEZE and/or STATUS PERMIT commands. However, read-only/update access to statuses is controlled by the Update and Read-only Window Types in each users Status Window.

The READ-ONLY and UPDATE keywords enable you (the dictionary Controller) to adjust the Status Window for Update and Read-only; other dictionary users cannot adjust these Window Types. You can control the read-only/update condition of a status for individual and/or groups of users by placing a STATUS WINDOW command, specifying a MAXIMUM for READ-ONLY and/or UPDATE as required, in Profiles and/or Corporate Executive Routines.

The default Window Range of the Status Window for Read-only and Update is DICTIONARY. This can only be adjusted by a STATUS WINDOW command in which MAXIMUM is specified.

An update status is accessed as an update status only if it is included in the Status Window for Update. If it is not included in the Status Window for Update but is included in the Status window for Read-only, it is accessed as a read-only status and appears as a read-only status in output from a STATUS LIST command.

Any status excluded from the Status Window for Update and for Read-only cannot be used as the current status. That is, it cannot be accessed through a STATUS status-name command. Output from a STATUS LIST command will indicate the read/write condition of each status as designated by STATUS FREEZE and/or STATUS PERMIT commands.

STATUS-LIST and NONE Options

The STATUS-LIST keyword enables you (the dictionary Controller) to adjust the Status Window for Status-list; other dictionary users cannot adjust this Window Type. You can control the information output to a user in response to STATUS LIST commands by placing a STATUS WINDOW command, specifying a MAXIMUM for READ-ONLY and/or UPDATE as required, in Profiles and/or Corporate Executive Routines.

A status must be included within the Status Window in order to be included in the output from a STATUS LIST command.

Thus, you can tailor the status environment so that the user need only be aware of the statuses with which he/she needs to work.

The default Window Range of the Status Window for Status-list is DICTIONARY. This can only be adjusted by a STATUS WINDOW command in which MAXIMUM is specified.

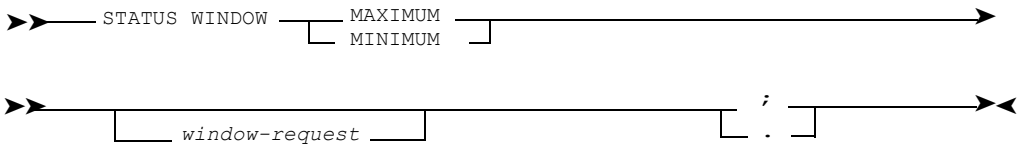
It is unlikely that you will want to suppress, from a user's STATUS LIST output, the direct or indirect base statuses of a status which is included in the output. However, if you do so, ControlManager automatically closes up any gaps that would appear in the output as the result of such a status being suppressed.

If the window-range keyword NONE is specified, no statuses are included in the Window Type(s) for which it is specified. Therefore, no statuses are processed for the activity to which that/those Window Type(s) relate.

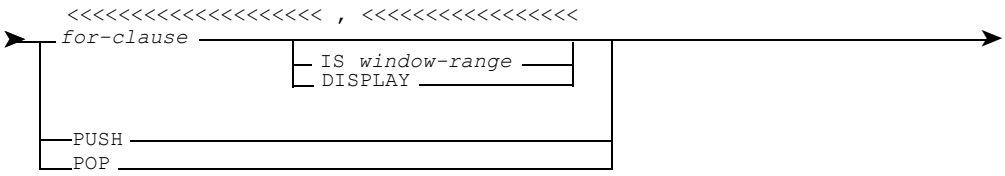
Refer to *ASG-Manager Products Basic Status* or *ASG-Manager Products Advanced Status* for details of the output from a STATUS LIST command.

See Chapter 8, "Status Facilities for Controllers," on page 53 for details of STATUS WINDOW MAXIMUM/MINIMUM in the general context of security by status.

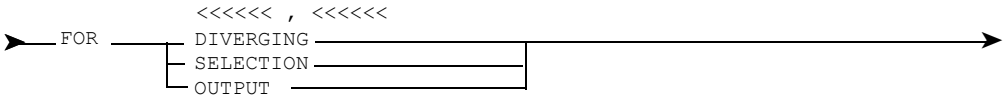
STATUS WINDOW Syntax



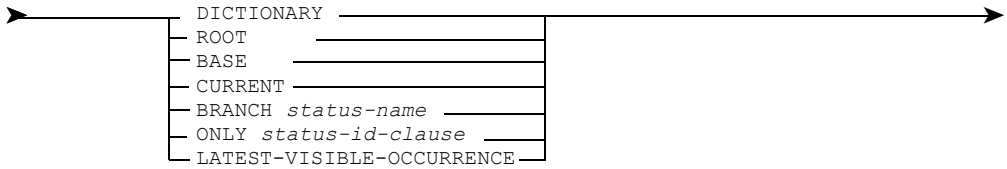
where *window-request* is:



where *for-clause* is:



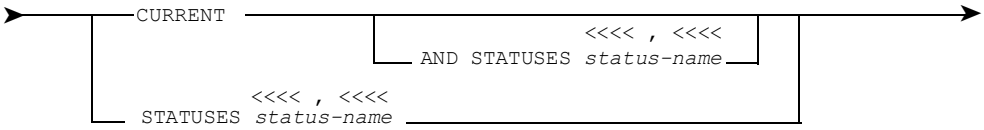
where *window range* is:



where:

status-name is the name of a status

status-id-clause is:



where *status-name* is as defined above.

Note:

If in the FOR clause you only specify SELECTION and/or OUTPUT, you may also specify LATEST-VISIBLE-OCCURRENCE as the window-range.

UNLOAD

The UNLOAD command outputs a physical copy of a dictionary to tape (or disk), as follows:

```
UNLOAD;
```

The output dataset from an UNLOAD command is intended only for subsequent use with a RELOAD command. Its format is such that it is of no use in any other context.

You should use the UNLOAD and RELOAD commands to complete these tasks:

- Make regular (ASG suggests daily or weekly) backup copies of a dictionary in case it becomes necessary to rebuild the dictionary. For example, if the dictionary datasets accidentally get overwritten.
- Change the space allocation of a BDAM or VSAM-organized dictionary.

You must use the Manager Products UNLOAD command to create a backup of a DIV dictionary created with a log. If you do not and it becomes necessary to rebuild the dictionary, then recovery of the dictionary from non-Manager Products backup will overwrite any transactions written to the log since the backup was taken. Roll-forward of the dictionary to its latest updated state is not possible.

The UNLOAD command is not accepted if the dictionary is open in read-only mode. If you open the dictionary in update mode, you ensure that any necessary recovery takes place before you UNLOAD the dictionary.

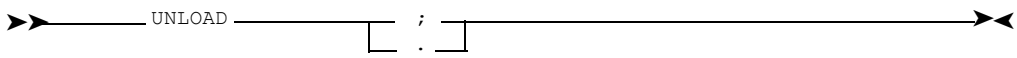
Only those blocks that have been utilized from the index, source, and data entries datasets are output. The recovery and log datasets are not output. (The automatic recovery system operates, if necessarily, on receipt of the UNLOAD command so that the recovery dataset is empty when you enter the UNLOAD command.)

In all environments the output dataset default block size is 9442 bytes. This value may be changed by using the optional DCB = BLKSIZE parameter with the UNLOAD DD statement for OS environments, or by using the optional BLKSIZE or BLOCK parameter with the UNLOAD FILEDEF statement for CMS. DOS users can use the UBLK keyword in the DCUST macro in order to change the block size.

In BS2000 environments the blocksize is determined from the job control. If the blocksize is not stated in the job control or is not sufficiently large, then it is calculated as the largest blocksize of the index, source, and data entries datasets, rounded up if necessary to a multiple of four, plus eight bytes.

The UNLOAD command is not available in CICS environments.

UNLOAD Syntax



XPRINT

It may sometimes be necessary for ASG to request a print-out of the internal formats of some parts of a repository in order to satisfy a request for maintenance. The XPRINT command allows you to obtain these selective printouts.

To obtain the printout, enter:

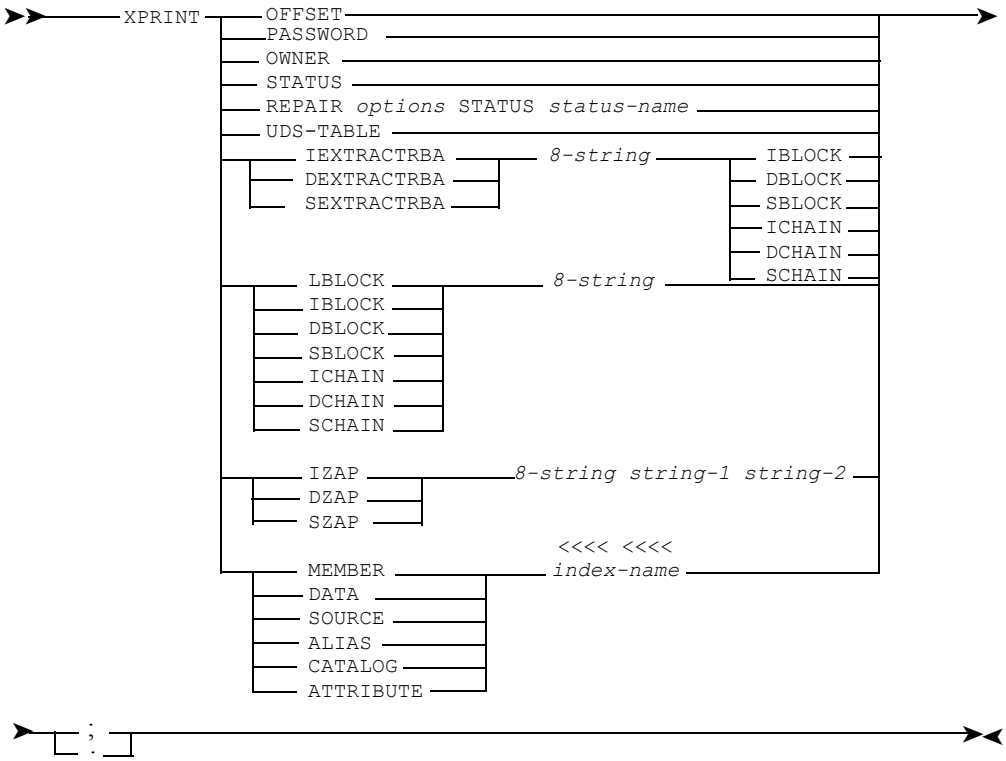
```
XPRINT details ;
```

where *details* is a keyword or clause that the ASG Service Desk has asked you to include in the command.

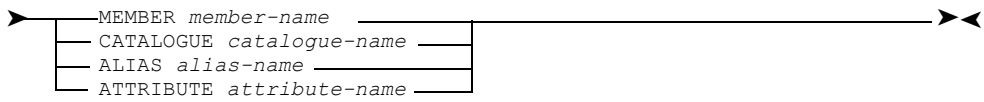
The output from the command is for ASG use only. No user documentation of the format of the output will be provided. The output from the command is identified by messages DM11546 through DM11552.

The ASG Service Desk will advise you as to which keyword or clause to use in this command on any occasion on which a printout is requested.

XPRINT Syntax



where *options* are:



Secondary Selection

Refer to *ASG-Manager Products Dictionary/Repository User's Guide* for further details.

Index